SC28-0912-0

OS/VS2 MVS TSO
3270 Extended Display Support - Session Manager
User's Guide and Reference

Program Product

IBM

SC28-0912-0

OS/VS2 MVS TSO
3270 Extended Display Support - Session Manager
User's Guide and Reference

Program Product

Program Number 5740-XE2

Version 1, Release 2

IBM

First Edition (July, 1977)

This edition with Technical Newsletter GN28-2928, applies to Release 2, Modification Level 0 (R2.0), of the program product TSO Session Manager (5740-XE2) and to all subsequent Releases until otherwise indicated in new editions or Technical Newsletters.

Changes are continually made to the information herein; before using this publication in connection with the operating of IBM systems, consult the latest IBM System/370 Bibliography, GC20-0001, for the editions that are application and current.

Publications are not stocked at the address given below; requests for IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for reader's comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, TSO Publications Development, Department D09, Building 706-2, PO Box 390, Poughkeepsie, NY 12602. Comments become the property of IBM.

PREFACE

This publication describes the facilities provided by the TSO Session Manager. Its purpose is to convey how-to, command syntax, and system programmer information required to use the Session Manager. The publication has six sections:

1. A Session Manager Primer

   introduces the use of TSO at a Session Manager-supported display terminal in the context of the IBM-supplied default display environment.

2. Using the Session Manager Commands

   give contextual examples of the use of the Session Manager commands.

3. The Session Manager Command Language

   explains the syntax of the Session Manager commands and describes each command.

4. The Default Display Environment

   provides details of the IBM-supplied default display environment, including the commands used to produce it.

5. System Programmer Reference

   provides information for system programmers who install and maintain the Session Manager.

6. Messages

   lists the messages issued by the Session Manager.

This publication's examples assume you are using an IBM 3270 Information Display Terminal, model 2.

Referenced Publications:

- OS/VS2 TSO Command Language Reference, GC28-0646

- OS/VS2 TSO Terminal User's Guide, GC28-0645
- TSO Command Language for Application Programmers Primer, SR20-4193-1
- Operator's Guide for IBM 3270 Information Display Systems, GA27-2742
- TSO 3270 Structured Programming Facility (SPF) Version 2 Program Reference Manual, SH20-1975.

CONTENTS

FIGURES

SUMMARY OF AMENDMENTS

Summary of Amendments
for SC28-0912-0
as Updated by TNL SN28-2928
TSO Session Manager Version 1, Release 2

Additions and changes have been made in this publication to reflect the following items:

| IBM 3276/3278 Display Terminals
| new support for larger screen sizes and 24 PF keys.

| Installation Management Exits
| new support of installation exit routines
| to allow monitoring of a TSO user's terminal session.

| Different Default Environments for Users
| new support allowing different default environments for different users.

INTRODUCTION

As a typical time sharing system, the TSO system, commands, and user programs are line-oriented. The TSO user enters commands to the system in a string syntax, and output resulting from these commands is placed by the system on succeeding lines of the terminal paper or screen. While these line-oriented programs and commands operate on both typewriter and display terminals, they do not utilize most of the features offered by display terminals.

The principal goal of the TSO Session Manager is to provide support that will extend all the facilities of the IBM 3270 Information Display Terminal to these programs and commands, and eliminate the operational problems associated with standard TSO display terminal support. Because this support extends across the TSO user's session and allows the user to better control his interactive conversation with the system, the name of "Session Manager" was chosen. The objectives of this support are to:

1. Increase the productivity of the TSO user.

   The Session Manager exploits the facilities of the display terminal to save time for the TSO user. Many features of the Session Manager work together to achieve this objective.

   - The user can control what data is displayed or not displayed at the terminal.

   - Unnecessary key strokes are avoided by allowing the user to form new input from data displayed on the screen. The Program Function keys of the IBM 3270 Display Terminal may be defined as often-used TSO commands, input sequences or commands to application programs, or Session Manager commands.

   - An unlocked keyboard allows data entry at all times. (This may be set by the user.)

   - Messages from other TSO users, the operator, or background jobs do not interrupt the user's session. They are retained in the session journal, and are not lost. The user may set the audible alarm to sound when a message is received.

   - The multitask structure of the Session Manager allows the user to execute Session Manager commands while TSO commands are being processed.

   - A journal of the entire line-oriented TSO session allows the user to review previous command input and output.

Today, a TSO user will often re-execute TSO commands just to have the output redisplayed, when this is possible (often it is not). This is avoided under the Session Manager.

- A "split screen" capability (not limited to a single "split") allows the user to view different data at the same time, reducing the need for printing data on system or local printers. For example, a Session Manager user could view the assembler listing of a program and test it at the same time.

2. Provide display support while allowing TSO commands and user-written programs to run unchanged.

   The Session Manager provides display support to existing line-oriented commands, programs, and CLISTs such that they do not have to be rewritten to take advantage of this support.

   Note: Exceptions to current TSO processing are listed below.

   When a WRITENR is issued in a CLIST, the cursor will no longer be positioned at the end of the text written to the screen. Instead, the cursor will return to the permanent cursor position to allow data entry.

   When entering TSO input data, the maximum line length cannot exceed 512 characters.

   Multiple TPUTS to a single line will now result in a single line for each TPUT.

   Password protected data sets will not be deleted unless the password is supplied correctly on the initial "DELETE" request. A prompt for the password is impossible in this situation and as a result, a possible system interlock may occur.

   The entire TSO command set is supported, including TSO TEST, TSO CLISTs, TSO EDIT, and any user-written commands and programs, with the exclusion of "full screen" programs (See Objective 6).

3. Functionally enhance TSO by fully utilizing the facilities of the IBM 3270 Display Terminal.

   The Session Manager utilizes the hardware features of the display terminal to make TSO easier to use, and allow the TSO user to be more productive. Session Manager use of the

display terminal includes these features:

- User-entered commands and other TSO input are displayed at high intensity, while system output is displayed at normal intensity. This helps the user distinguish the two sides of the interactive conversation. When the session journal (the TSOOUT stream) is copied to a system printer, the highlighted lines are overprinted to appear darker, thereby extending this feature to the hardcopy listing of the session.

- The TSO command or program currently executing is displayed at high intensity in the input-only journal (the TSOIN stream). This enables the user to keep track of which command is executing provided a window is viewing the TSOIN stream.

- The audible alarm (if present) may be set by the user to indicate that:

  - a message from another user or a background job has been received.

  - the keyboard is unlocked and ready for input.

  - new data has been displayed on the screen.

  - a Session Manager error message has been issued.

  - a TSO command or program has begun executing.

- The Program Function keys (if present) may be defined as TSO commands or subcommands, Session Manager commands, input to application programs, or other useful data. Symbolic substitution of data entered from the keyboard may be combined with the Program Function key definition.

4. Provide session-wide system-level support.

   Session Manager display support is available to the TSO user from LOGON to LOGOFF. It covers all TSO commands and modes (except "full screen" programs), as well as any programs running under TSO.

5. Retain the capabilities the TSO user has on a typewriter terminal.

   The Session Manager provides the display terminal user with the functions the typewriter user has such as an unlocked keyboard, retention of messages from other TSO users, hardcopy of the session, etc.

6. Co-exist with "full-screen" applications.

Some TSO commands and programs are written to run specifically on the IBM 3270 Display Terminal ("full screen" programs). The Structured Programming Facility, an IBM TSO Program Product, is an example of such a command. The Session Manager "steps aside" when these display-oriented programs are executing, allowing them complete control of the display terminal. While such full screen programs are running, all of the Session Manager functions are disabled, except the handling of messages from other TSO users, background jobs and the operator. Copies of these messages are placed in the session journal even though the full screen application retains control of the screen.

7. Give the TSO user control of the display terminal.

The Session Manager command language allows the TSO user to dynamically redefine part or all of his display environment at any time during the TSO session. None of the features of the display terminal (certain lines of the screen, certain Program Function keys, etc.) are reserved for use by the Session Manager. Thus, the TSO user can tailor the operation of the terminal to his needs and tastes, in effect "customizing" the terminal to the task at hand.

1. A SESSION MANAGER PRIMER

Introduction

This primer will introduce you to using TSO at the IBM 3270 Display Terminal as supported by the TSO Session Manager. Some basic concepts are introduced, and the use of the IBM-supplied default screen layout and Program Function key definitions is explained. The use of the Session Manager commands is not explained in this section; that is beyond the scope of a primer. See "Using the Session Manager Commands", for an introduction to the Session Manager commands.

You are assumed to be familiar with TSO and to know how to operate the IBM 3270 Display Terminal. If you are not, it is strongly suggested that you read:

- TSO Command Language for Application Programmers Primer, SR20-4193. This will introduce you to using TSO to write and debug application programs.
- OS/VS2 TSO Terminal User's Guide, GC28-0645. This will introduce you to the facilities and usage conventions of TSO.
- Operator's Guide for IBM 3270 Information Display Systems, GA27-2742. This will introduce you to the use of the 3270, including the use the various keyboards available.
- OS/VS2 TSO Command Language Reference, GC28-0646. This provides a description of each of the TSO commands.

This primer can best be used if you browse through it first and then sit down at a TSO terminal and go through the examples described. The illustrations show what one user's screen looked like as he went through the primer. Your screen will appear similar, but it won't be exactly the same due to variations in each TSO installation and the different data sets that will be in your catalog.

This primer assumes the IBM-supplied default screen layout for a 3270 Model 2 Display Terminal (i.e., 24 lines) and twelve PF key definitions.

The IBM 3270 Keyboard

The Session Manager assigns new meanings to some of the keys of the 3270 terminal keyboard. Figure 1.1 illustrates one of the 3270 terminal keyboards. See the "Operators Guide for IBM 3270 Information Display Systems" for complete descriptions of all the 3270 terminal keyboards. The keys whose meanings are different from those described in the Operator's Guide are described next:

CLEAR

The CLEAR key is used to initiate entry of Session Manager commands. Pressing it clears the screen and displays a message inviting you to enter a Session Manager command. Typing a command and pressing the ENTER key or any PF key will send the command to the Session Manager. For example, you could type:

```
reset
```

and press the ENTER key to have the default screen layout re-established. The Program Function keys do not have their normal meanings here; they act the same as the ENTER key. After you have entered a command, these keys return to their normal definitions.

ERASE INPUT

The ERASE INPUT key erases all the areas of the screen where you may enter input. The "protected" areas of the screen (where you may not type) are not erased.

ERASE EOF

The ERASE EOF key erases all the data in one line to the right of the cursor's location. Its principal use is for generating a "null line" for TSO. This is described later in the Primer.

TEST REQ

The TEST REQ key ("Test Request") is used to enter Session Manager commands. If you type data anywhere on the screen and then press the TEST REQ key, the data will be transmitted to the Session Manager and interpreted as a command.

Note: The TEST REQ may be reserved for other purposes at your installation. Check with your system programmer before pressing it.

PA1

The PA1 (Attention) key is used to generate an attention in-

terrupt to a TSO command or program. Pressing it does not interrupt the Session Manager.

DUP

The DUP (Duplicate) key is not supported by the Session Manager.

PA2

The PA2 key (labeled "CNCL" on some keyboards) is used to refresh the screen. Pressing it causes the Session Manager to redisplay the entire screen. Any data you typed immediately before pressing the PA2 key is lost.

FIELD MARK

The FIELD MARK key is used to separate TSO commands when multiple commands are entered on one line. When this key is pressed, a ";" is displayed. (If PCF II is installed then two Field Mark keys are required to separate TSO commands within a Clist.)

ENTER

Pressing the ENTER key causes the Session Manager to read the data you have typed on the screen. In the IBM-supplied default screen layout, the data is given to TSO to be interpreted as commands or other input. The display screen is then updated by the Session Manager if any new data need to be displayed.

PF1 - PF12 (not illustrated in Figure 1.1)

The twelve Program Function (PF) keys may be defined by the CHANGE.PFK Session Manager command (see "Environment Definition Commands"). The default PF key definitions are described later in this primer.

The other special keys of the 3270 keyboard are referred to in this primer as "terminal editing keys'. They are described in the "Operator's Guide for IBM 3270 Information Display Systems, GA27-2742."

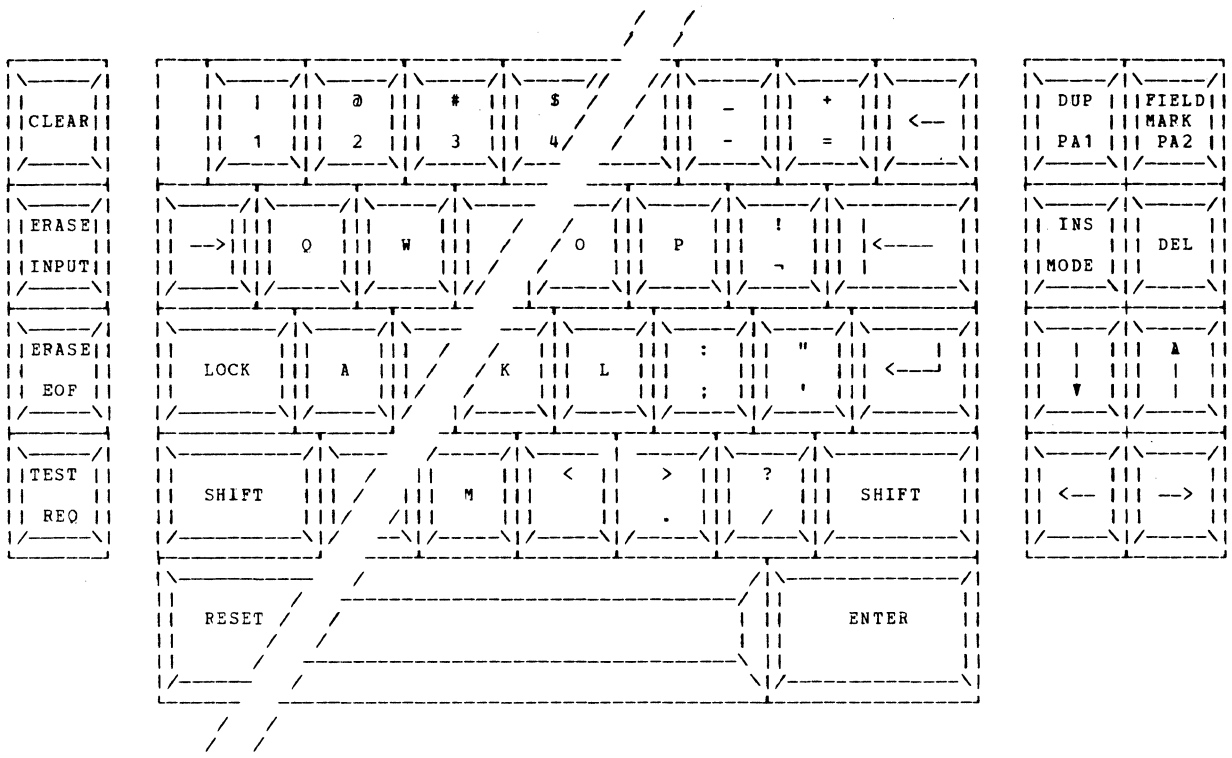


Figure 1.1 - a figure showing the 3270 keyboard

Logging On to TSO

Logging on to TSO at a Session Manager-supported terminal is similar to logging on to TSO without the Session Manager. You may need to specify a special logon procedure name. Your system programmer will provide you the procedure name if one is needed. Figure 1.2 shows a user's screen after he logged on to TSO, using a Logon Procedure of "SMNEW". Because he did not specify a password in the LOGON command, TSO prompted him to enter it. Note that the password is not displayed when entered in response to the prompting message.

After you have logged on, TSO Broadcast Messages may be displayed. Then the Session Manager will display a message inviting you to begin your TSO session, as shown in Figure 1.2. Press the ENTER key to begin your TSO session. Pressing the ENTER key in this manual will be shown as:

[ENTER]

The pressing of other keys will be similarly illustrated.

It's important to remember that during your TSO session you are communicating with two different programs. The first is TSO, which "does the work" - editing, compiling and running programs, etc. The second is the Session Manager, which helps you control what is displayed at your terminal. In this primer you will enter commands to TSO and use the Program Function keys to tell the Session Manager what data you want displayed on the screen.

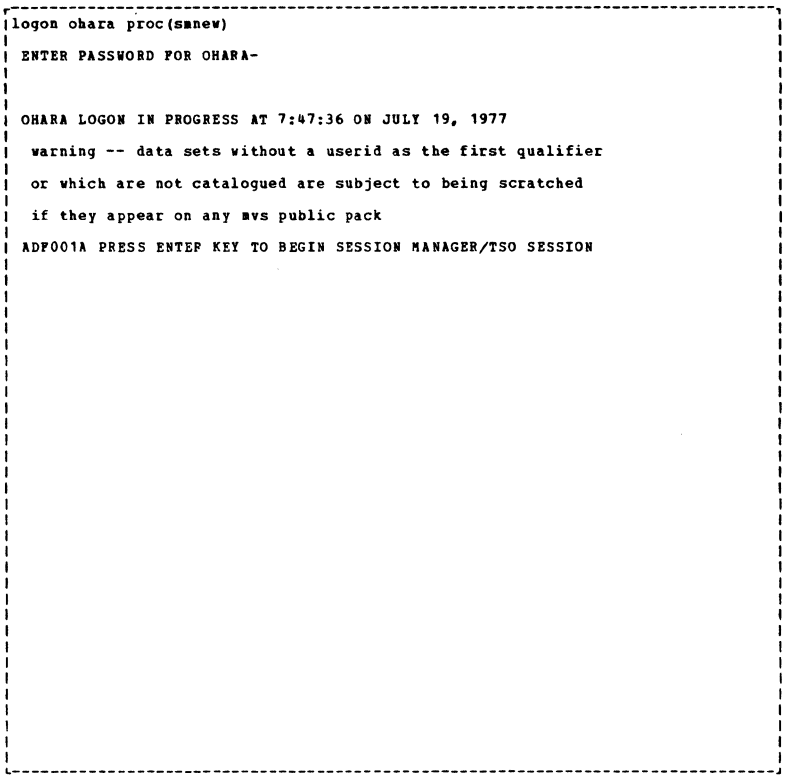


Figure 1.2 - Logging on to TSO

Use of the ENTER and PA2 Keys

Your display screen should now look similar to Figure 1.3. You will find the cursor in the lower left hand corner of the display screen. Directly above the cursor should be the TSO "READY" message. If "READY" is not visible, press:

[ENTER]

and it will appear. TSO is now ready for you to enter commands.

One of the differences between using TSO at a regular terminal

and at a Session Manager supported terminal is the use of the ENTER key. Press:

[ENTER]

and notice what happens.

Nothing happens. You can press it several times and nothing will happen. Why? The ENTER key when pressed alone does not send a message to TSO, it only tells the Session Manager you wish the screen to be updated if there is new data to be displayed. Since there is no new data to be displayed at this time, the Session Manager does nothing. Pressing the ENTER key alone is not the same as pressing the Carriage Return on a typewriter terminal or at a display terminal without the Session Manager. The ERASE EOF key must be pressed <u>before</u> pressing the ENTER key in order to transmit a "null line".

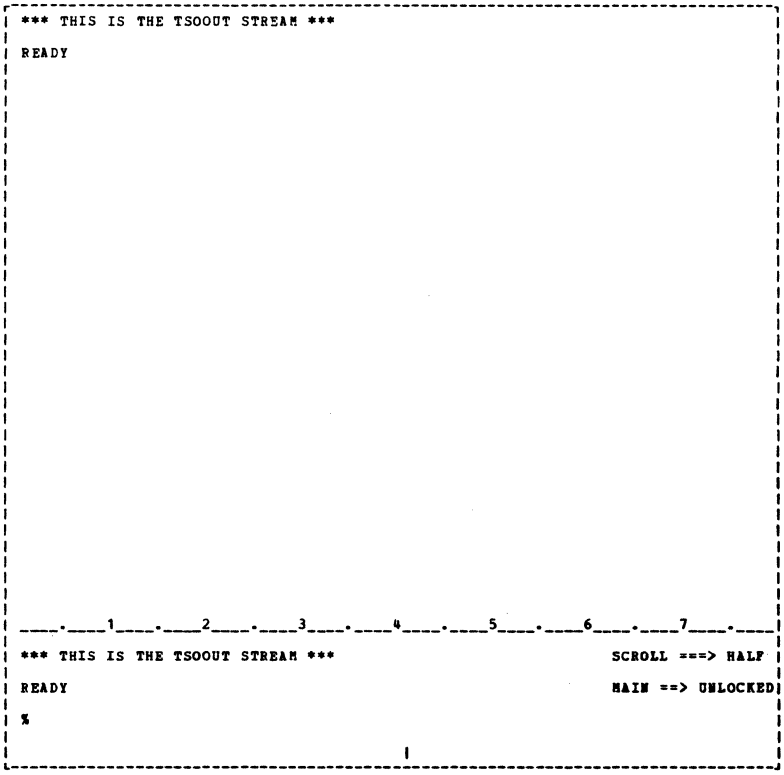


Figure 1.3 - The Default Session Manager Screen Layout

If you wish to correct a mistake in a line being entered, you can use the terminal editing keys to correct the error before you press the ENTER key. Another way is to cancel the data you have entered by pressing the PA2 key (labeled "CNCL" on some keyboards and shown in this primer as "[PA2]"). For example, type a line of data on the terminal, but don't press the ENTER key. Press:

[PA2]

and you will see the line has been erased and replaced with what was there before. No data has been transmitted to TSO.

To execute the TSO command "TIME", enter:

time [ENTER]

(Type "time" and press the ENTER key.) Your screen should now look like Figure 1.4.

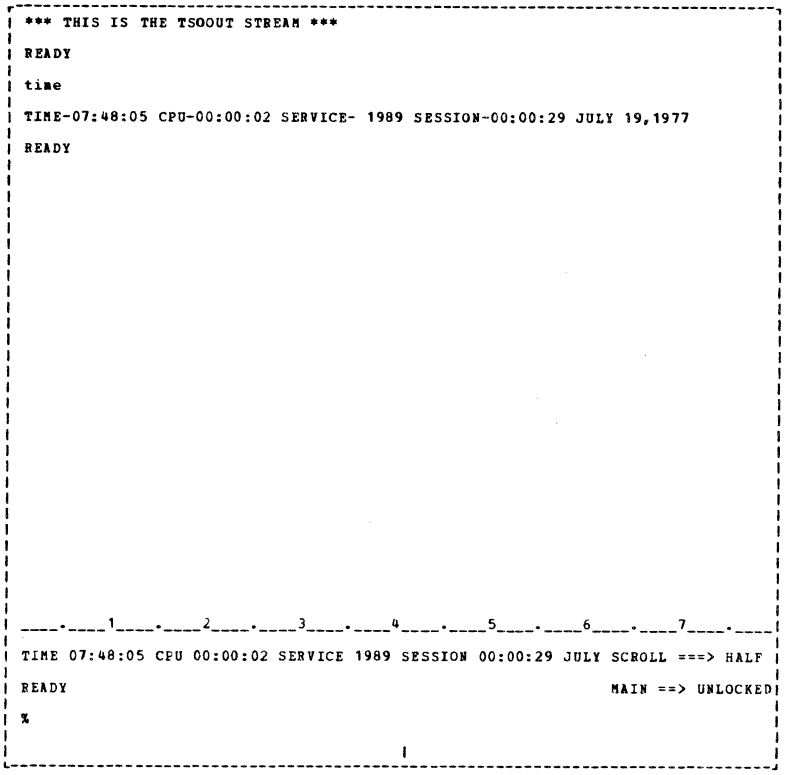


Figure 1.4 - The TIME Command Has Been Entered

Session Manager Streams

If you have used TSO at a typewriter terminal (the IBM 2741 Communications Terminal, for example), you will remember that after you had been logged on for a while you had produced a long sheet of paper. This sheet of paper was a record of your TSO session: everything you typed (commands and input to TSO) and everything the terminal typed (output from commands, messages, etc.). Such

printed listings of your TSO session are unavailable at a display terminal; therefore the Session Manager records your input to TSO and TSO's output to you. This collection of saved lines (a journal of your TSO session) is called a stream. The name of the stream containing your TSO input and output is "TSOOUT." Your input and TSO's output are interleaved in the TSOOUT stream, just as they were on the typewriter terminal listing.

Streams (other streams will be introduced later) have a "top" and "bottom". Each line of data entering a stream is placed in the next available line of the stream, starting at the top. Just as the data at the top of the typewriter terminal sheet is printed before the data at the bottom and thus is the "oldest" data on the paper, the data at the top of a stream is considered to be the oldest data in the stream. The most recent line of data to be placed in the stream is the "newest" data and is considered to be the bottom of the stream. The streams, whose capacities are fixed when they are defined (by default or by installation modification), fill up with data as your session progresses. When the stream's capacity is exceeded, the stream "wraps around", and new data entering the stream overlays the oldest data at the top of the stream, causing the overlayed data to be lost. The "top" of the stream is pushed down as new data enters, but always contains the oldest data in the stream.

The data displayed at the top of your screen is the data presently in the TSOOUT stream. This is shown in Figure 1.4. The first line of the stream is a title line, identifying this stream as the TSOOUT stream. The next line is the first READY message that TSO issued, inviting you to enter a command. Next, is the TIME command you entered. Notice that it is highlighted (displayed at a brighter intensity). Everything you type during the TSO session will be highlighted, to help you to distinguish it from TSO's output. The next line is the output line from the TIME command, followed by another READY message. Since the READY message is the last line in the stream, you know that TSO is expecting you to enter another command.

By using the Session Manager-defined Program Function keys, you can view the most recent data of your session (as in normal TSO on a 3270), or you can view previous input and output from the session.

Session Manager Windows

The display screen is divided into areas which are called windows. You view the data of your session (in a stream) through these windows on the display screen. Two windows will be introduced to you at this time. They are separated by the dashed line in the lower portion of the screen. The window above the dashed line is called the "MAIN" window. It displays 19 lines of data,

with each line displaying up to 79 characters. Figure 1.5 illustrates how the window named "MAIN" is logically positioned over the TSOOUT stream, displaying the first 19 lines of data in the stream.

The window just below the dashed line is called the CURRENT window. It displays the last two lines of the TSOOUT stream (the newest data in the stream), with each line displaying up to 62 characters.

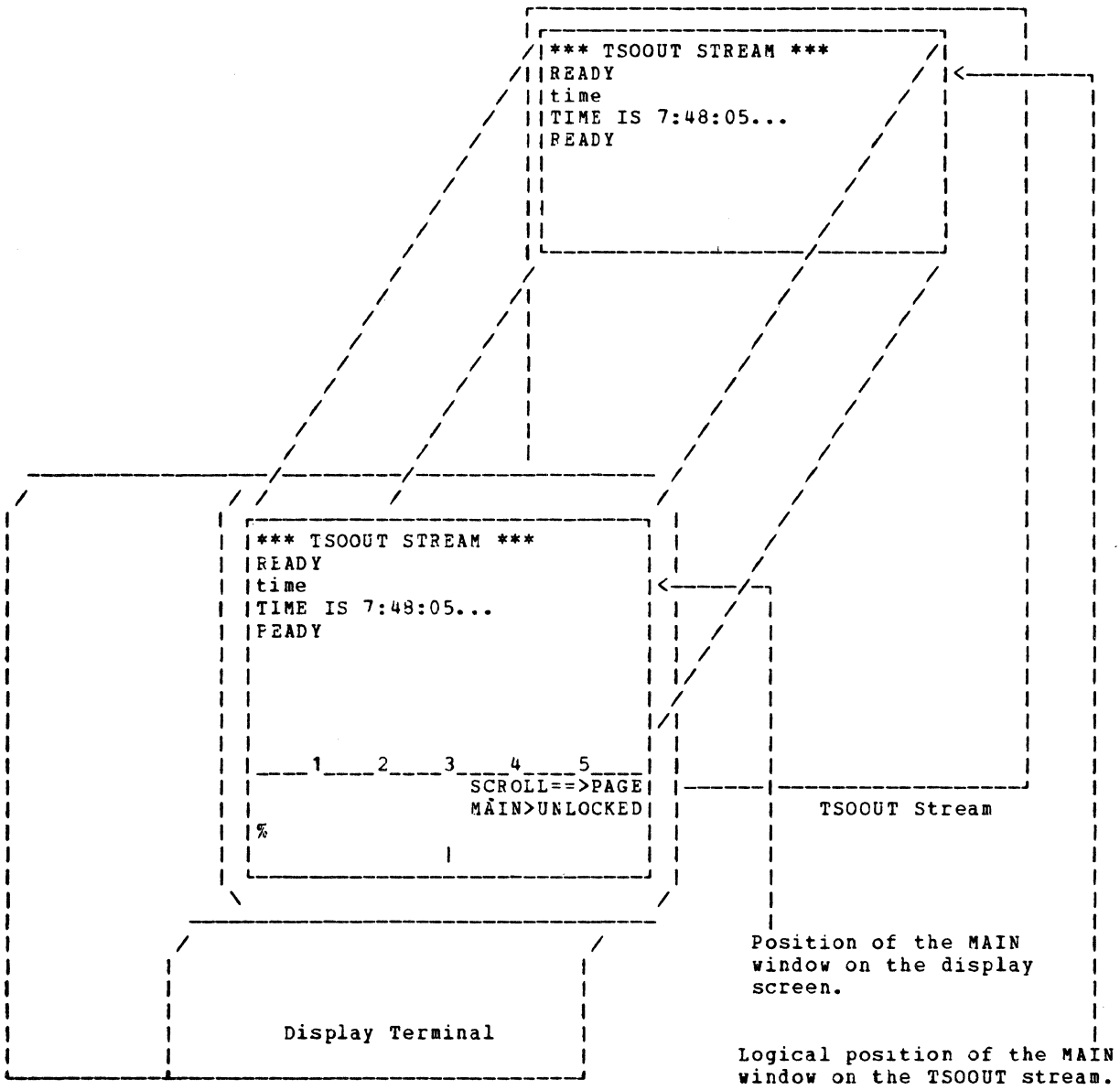


Figure 1.5 - Streams and Windows

Controlling the Session Manager Windows

Now let's explore how you can use the Session Manager to control the data displayed on the screen. To create some data to display, enter:

help protect [ENTER]

Your screen should now look like Figure 1.6. If you don't see the READY message in the MAIN window, press the ENTER key again (to tell the Session Manager to update the screen) and it will appear.

Note: The need to press the enter key additional times is dependent on the response time of TSO at your installation. Some TSO commands and programs take more time to execute than others. If the Session Manager updates the display screen before the command has produced any output, the output will not be displayed. Pressing the ENTER key again tells the Session Manager to update the display screen if any new data has yet to be displayed.

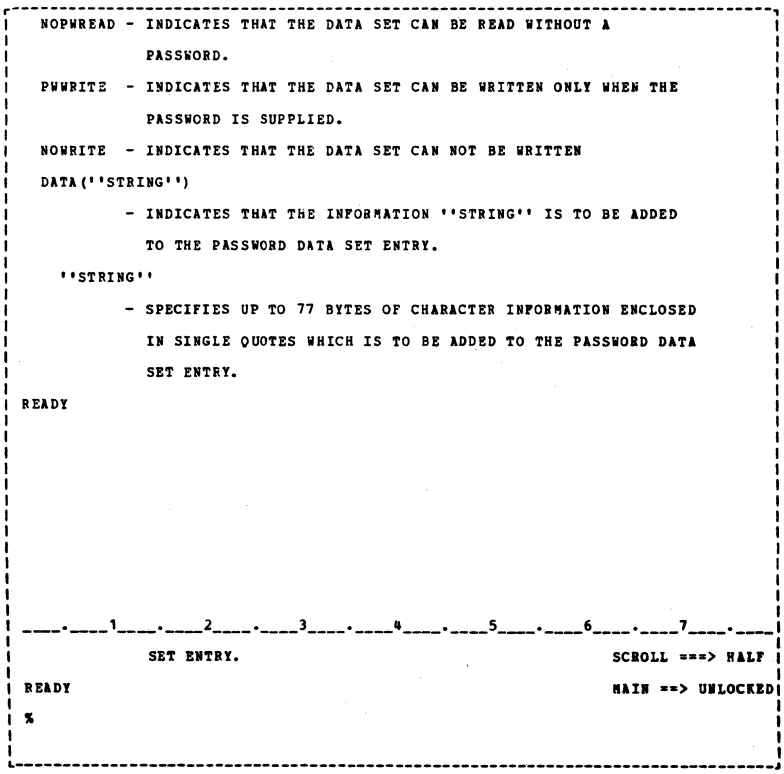


Figure 1.6 - The HELP Command Has Been Entered

What has just happened? An analogy can be drawn with a typewriter terminal. Every time TSO types a new line at a typewriter, the carriage moves the paper up one space to make room for the line. Thus the carriage is always at the newest line on the paper. In the same fashion, as new lines are placed by TSO in the Session Manager's TSOOUT stream, the Session Manager moves the window to display the new data in the stream, updating your screen in the same fashion as the typewriter.

Scrolling

At the typewriter, if you wish to review the output of your session, you merely reach forward and pull the paper back to see it. At a Session Manager-supported display terminal, you use Program Function (PF) keys to tell the Session Manager to move the MAIN window over the TSOOUT stream to view the data in the stream. The Function keys and the operations they perform are listed below and shown in Figure 1.7.

Note: The PF keys are shown reduced in size for use on your terminal keyboard in "Appendix A: Default PF Key Definitions".

Key Operation

PF 1 Print the stream containing the snapshots of the screen (taken by pressing PF 4) on the system printer, and erase the contents of the stream, making room for new snapshots.

Note: This PF key generates a TSO command to print the stream. You can only press this key when you are in TSO command mode.

PF 2 Set the scroll value for the scrolling PF keys (7,8,10,11). You may type "HALF", "PAGE", "MAX", or an integer specifying the number of lines to scroll. This value is shown in the scroll amount field on the display screen.

PF 3 Enter Session Manager commands. The use of this PF key is explained in "Using the Session Manager Commands".

PF 4 Take a "Snapshot" of the screen image and place it in a Session Manager stream. This stream (named "EXTRA1") has a capacity of 6 snapshots.

PF 5 Scroll the MAIN window backwards until the entered text string is found.

PF 6 Flip-flops the CURRENT window between viewing the TSOIN and TSOOUT streams.

PF 7 Scroll the MAIN window backward the number of lines shown in the scroll amount field.

PF 8 Scroll the MAIN window forward the number of lines shown in the scroll amount field.

PF 9 Scroll the MAIN window backward to the oldest data.

PF 10 Scroll the MAIN window left the number of columns shown in the scroll amount field.

PF 11 Scroll the MAIN window right the number of columns shown in the scroll amount field.

PF 12 Move the MAIN window to the newest data in the stream and unlock it.

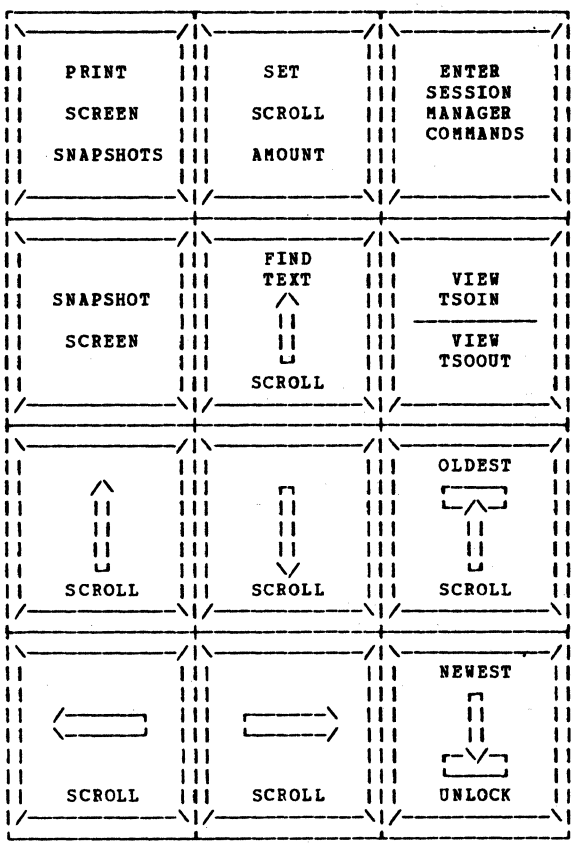


Figure 1.7 - Program Function Key Definitions

The process of moving a window to view different data in the stream is called scrolling. The PF keys which perform the scrolling operations are labeled "SCROLL" in Figure 1.7. Four of the scrolling PF keys (PF 7,8,10,11) move the window a specific amount. This amount is displayed in the scroll amount field (labeled "SCROLL ===>" on the display screen), and is set by typing

a value and pressing PF 2. (Only the first five characters of the scroll amount are displayed.) Valid scroll amounts are:

- An integer from -99999 to 999999: specifies the number of lines to be scrolled when PF 7 or PF 8 are pressed.

  An integer from -32767 to 32767: specifies the number of columns to be scrolled when PF 10 or PF 11 are pressed.

  Note: If an out-of-range value is specified, it will be rounded to the nearest acceptable value when the scrolling key is pressed.

  If a negative number is specified the scrolling direction will be reversed.

- PAGE: specifies scrolling by one page, a page being the number of lines or columns in the window.

- HALF: specifies scrolling by one half page.

- MAX: specifies scrolling to the top (oldest data), bottom (newest data), left edge (column 1), or right edge (column 32768), depending on which PF is pressed.

PAGE, HALF and MAX may be abbreviated "P", "H", and "M", respectively. The scroll amount is set to HALF initially.

If you enter an invalid value and press PF 2, or enter no value and just press PF 2, no error message will be issued until you attempt to use the value by pressing one of the scrolling PF keys. For example, enter:

xxx [PF2]

"XXX" will now be displayed in the scroll amount field. Press:

[PF8]

and your screen should look like Figure 1.8. Notice the error message displayed in the lower right corner of the screen. The audible alarm (if present) will sound when the message is displayed.

```
   NOPWREAD - INDICATES THAT THE DATA SET CAN BE READ WITHOUT A
              PASSWORD.
   PWWRITE  - INDICATES THAT THE DATA SET CAN BE WRITTEN ONLY WHEN THE
              PASSWORD IS SUPPLIED.
   NOWRITE  - INDICATES THAT THE DATA SET CAN NOT BE WRITTEN
   DATA(''STRING'')
            - INDICATES THAT THE INFORMATION ''STRING'' IS TO BE ADDED
              TO THE PASSWORD DATA SET ENTRY.
     ''STRING''
            - SPECIFIES UP TO 77 BYTES OF CHARACTER INFORMATION ENCLOSED
              IN SINGLE QUOTES WHICH IS TO BE ADDED TO THE PASSWORD DATA
              SET ENTRY.
READY
____.____1____.____2____.____3____.____4____.____5____.____6____.____7____.____
              SET ENTRY.                                  SCROLL ===> XXX
READY                                                     MAIN ==> LOCKED
%
                                    | ADF017I INVALID SCROLL AMOUNT: XXX
```

Figure 1.8 - Pressing a PF Key With an Invalid Scroll Amount

To correct the scroll amount, enter:

page [PF2]

Notice that the error message is no longer displayed.

Pressing PF 7 will cause the MAIN window to scroll backward the amount displayed in the scroll amount field. To review the output of the HELP command, press:

[PF7]

and your screen should look like Figure 1.9.

```
    'PASSWORD1'
          - IS THE OLD OR CURRENT PASSWORD FOR THIS PASSWORD DATA SET
            ENTRY
    'PASSWORD2'
          - IS THE NEW PASSWORD FOR THIS PASSWORD DATA SET ENTRY
  DELETE('PASSWORD1')
          - SPECIFIES THAT AN ENTRY FOR THE DATA SET IS TO BE REMOVED
            FROM THE PASSWORD DATA SET.
    'PASSWORD1'
          - IS THE OLD OR CURRENT PASSWORD FOR THIS PASSWORD DATA SET
            ENTRY
  LIST('PASSWORD1')
          - SPECIFIES THAT THE SECURITY COUNTER, ACCESS TYPE, AND THE
            USER DATA FOR THIS ENTRY ARE TO BE LISTED.
    'PASSWORD1'
          - IS THE OLD OR CURRENT PASSWORD FOR THIS PASSWORD DATA SET
            ENTRY.
  PWREAD  - INDICATES THAT THE DATA SET CAN BE READ ONLY WHEN THE
            PASSWORD IS SUPPLIED.
____.____1____.____2____.____3____.____4____.____5____.____6____.____7____.____
            SET ENTRY.                                  SCROLL ===> PAGE
READY                                                   MAIN ==> LOCKED
%
```

Figure 1.9 - PF 7 Has Been Pressed

Press:

[PF7]

again, and the window will scroll back another page. Press:

[PF7]

twice more and you will scroll all the way back to the oldest data in the stream. The oldest data in the stream (the top of the stream) is the limit for scrolling backward. Your screen should now look like Figure 1.10. You can see the TIME command you typed followed by its output, and the HELP command you typed followed by the beginning of its output.

You can scroll directly to the oldest data from any position in the stream by pressing:

[PF9]

This scrolling key will always move the MAIN window to the oldest data in the TSOOUT stream. If the stream has not been filled to capacity, pressing PF 9 will cause the window to display the first data of your session, as shown in Figure 1.10. If the stream has filled to capacity, PF 9 will scroll the MAIN window

to the oldest data that has not been overlaid by the new data entering the stream.

```
*** THIS IS THE TSOOUT STREAM ***
READY
time
TIME-07:48:05 CPU-00:00:02 SERVICE-1989 SESSION-00:00:29 JULY 19,1977
READY
help protect

FUNCTION -
  THE PROTECT COMMAND IS USED TO CREATE, MODIFY, OR REMOVE THE
  PROTECTION ATTRIBUTES OF NEW AND OLD DATA SETS.

SYNTAX -
         PROTECT   'DSNAME'  ADD('PASSWORD2')/
                             REPLACE('PASSWORD1','PASSWORD2')/
                             DELETE('PASSWORD1')/
                             LIST('PASSWORD1')
                             PWREAD/NOPWREAD  PWWRITE/NOWRITE
                             DATA(''STRING'')
  REQUIRED - 'DSNAME'
____.____1____.____2____.____3____.____4____.____5____.____6____.____7____.____
             SET ENTRY.                                  SCROLL ===> PAGE
READY                                                    MAIN ==> LOCKED
%
```

Figure 1.10 - Scrolling to the Oldest Data

You can scroll the MAIN window forward in the same manner you scrolled it backward by pressing:

[PF8]

If you press it several more times, you will eventually reach the newest data in the stream (the bottom of the stream), as shown if Figure 1.8. This is the limit for forward scrolling, just as the oldest data in the stream is the limit for backward scrolling.

To set the scroll amount to HALF, enter:

half [PF2]

Pressing PF 7 or PF 8 will scroll the MAIN window backward or forward one half page. To set the scroll amount to 30 lines, enter:

30 [PF2]

Pressing PF 7 or PF 8 will now scroll the MAIN window backward or

forward 30 lines.

You can also scroll the MAIN window in a different manner: moving it until a specific group of characters has appeared in the window. This is accomplished by typing the characters to be scrolled to and then pressing PF 5. For example, to scroll back to where you entered the HELP command, enter:

help [PF5]

The Session Manager will use the text you just entered (called the "search argument") and search backward through the stream until it finds the search argument you entered in the stream. It will then scroll the MAIN window to that location, until the search argument is the top line displayed in the window. The result is shown in Figure 1.11.

Note: If the text you want to scroll to is displayed in uppercase letters, enter the search argument in uppercase before pressing PF 5.

If the search argument is not found, the window is not moved, a message is displayed in the lower right hand corner of the screen, and the audible alarm is sounded.

```
help protect


FUNCTION -
  THE PROTECT COMMAND IS USED TO CREATE, MODIFY, OR REMOVE THE
  PROTECTION ATTRIBUTES OF NEW AND OLD DATA SETS.


SYNTAX -
        PROTECT   'DSNAME'  ADD('PASSWORD2')/
                            REPLACE('PASSWORD1','PASSWORD2')/
                            DELETE('PASSWORD1')/
                            LIST('PASSWORD1').
                            PWREAD/NOPWREAD  PWWRITE/NOWRITE
                            DATA(''STRING'')
  REQUIRED - 'DSNAME'
  DEFAULTS - ADD


OPERANDS -
  'DSNAME' - NAME OF DATA SET TO BE PROTECTED.
  ADD('PASSWORD2')
____.____1____.____2____.____3____.____4____.____5____.____6____.____7____.____
            SET ENTRY.                                  SCROLL ===> 30
READY                                                   MAIN ==> LOCKED
%
```

Figure 1.11 - PF 5 Has Been Pressed

Locked and Unlocked Windows

To cause some new data to be placed in the TSOOUT stream, enter:

profile list [ENTER]

Now, to tell the Session Manager to update the screen, press:

[ENTER]

Your screen should look like Figure 1.12. If it doesn't, press the ENTER key again to make sure the Session Manager has updated the screen. Notice that the output from the PROFILE command is displayed in the CURRENT window, while the MAIN window is still at the top of the stream. This is because the MAIN window is locked, that is, frozen in position. The window, while locked, will not automatically move to display new data when it enters the stream. Whenever you press any of the scrolling PF keys, after the window has scrolled, the window will be locked in place and will no longer be updated to display the new data that enters the stream. However, this does not prevent new data from entering the bottom of the stream. The CURRENT window, however, is always unlocked, and thus it moved to display the output from the PROFILE command. An unlocked window always moves to display new data when it enters the stream.

Again an analogy can be drawn with a typewriter terminal. When you reach forward and pull back the paper to view the previous data of your session, you can no longer see the carriage typing new data. (And of course pulling back the page over the typewriter doesn't prevent new data from being typed.)

The locked/unlocked status of the MAIN window is displayed on the screen below the scroll amount field. Whenever you press one of the scrolling PF keys "LOCKED" will be displayed.

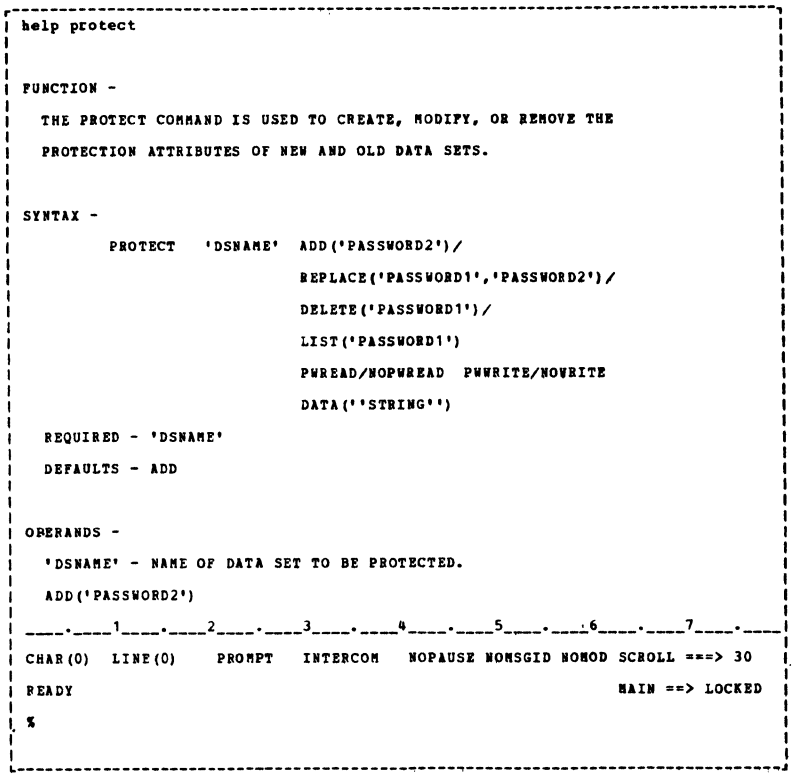


Figure 1.12 - The PROFILE Command Has Been Entered

Pressing PF 12 will unlock the MAIN window and move it to view the newest data in the stream. Press:

[PF12]

and your screen should appear as in Figure 1.13. Observe that "UNLOCKED" is now displayed below the scroll amount field. The output from the PROFILE command is displayed in the MAIN window. The Session Manager will now automatically move the MAIN window to display new data when it enters the TSOOUT stream.

```
            ENTRY.
  PWREAD   - INDICATES THAT THE DATA SET CAN BE READ ONLY WHEN THE
             PASSWORD IS SUPPLIED.
  NOPWREAD - INDICATES THAT THE DATA SET CAN BE READ WITHOUT A
             PASSWORD.
  PWWRITE  - INDICATES THAT THE DATA SET CAN BE WRITTEN ONLY WHEN THE
             PASSWORD IS SUPPLIED.
  NOWRITE  - INDICATES THAT THE DATA SET CAN NOT BE WRITTEN
  DATA(''STRING'')
           - INDICATES THAT THE INFORMATION ''STRING'' IS TO BE ADDED
             TO THE PASSWORD DATA SET ENTRY.
    ''STRING''
           - SPECIFIES UP TO 77 BYTES OF CHARACTER INFORMATION ENCLOSED
             IN SINGLE QUOTES WHICH IS TO BE ADDED TO THE PASSWORD DATA
             SET ENTRY.
READY
profile list
CHAR(0)  LINE(0)    PROMPT   INTERCOM   NOPAUSE NOMSGID NOMODE  NOWTPMSG PREFIX
READY
____.____1____.____2____.____3____.____4____.____5____.____6____.____7____.____
CHAR(0)  LINE(0)    PROMPT   INTERCOM   NOPAUSE NOMSGID NOMOD SCROLL ===> 30
READY                                                      MAIN ==> UNLOCKED
%
                                    |
```

Figure 1.13 - PF 12 Has Been Pressed

You will notice that the output from the PROFILE command has been truncated; the output from the command is wider than the screen. To view the rest of the output, you need to move the window to the right so as to have the truncated data displayed on the screen. Pressing PF 11 will move the MAIN window to the right the number columns shown in the scroll amount field (presently "30"). Press:

[PF11]

and your screen should look like Figure 1.14. Notice that the numbered, dashed line also moved 30 columns, in synchronization with the MAIN window. You can continue pressing PF 11 and each time the window will scroll an additional 30 columns to the right. However, eventually there will be no more data to display so you will be only viewing blank lines. To scroll to the left, press:

[PF10]

and the MAIN window will scroll to the left the number of columns shown in the scroll amount field. Press it as many times as you pressed PF 11 and your screen should again look like Figure 1.13, except that "LOCKED" is displayed below the scroll amount field.

Note: The dashed line only extends to column 240. If you continue to scroll to the right, you will eventually pass it. Don't be alarmed; the MAIN window is still scrolling to the right.

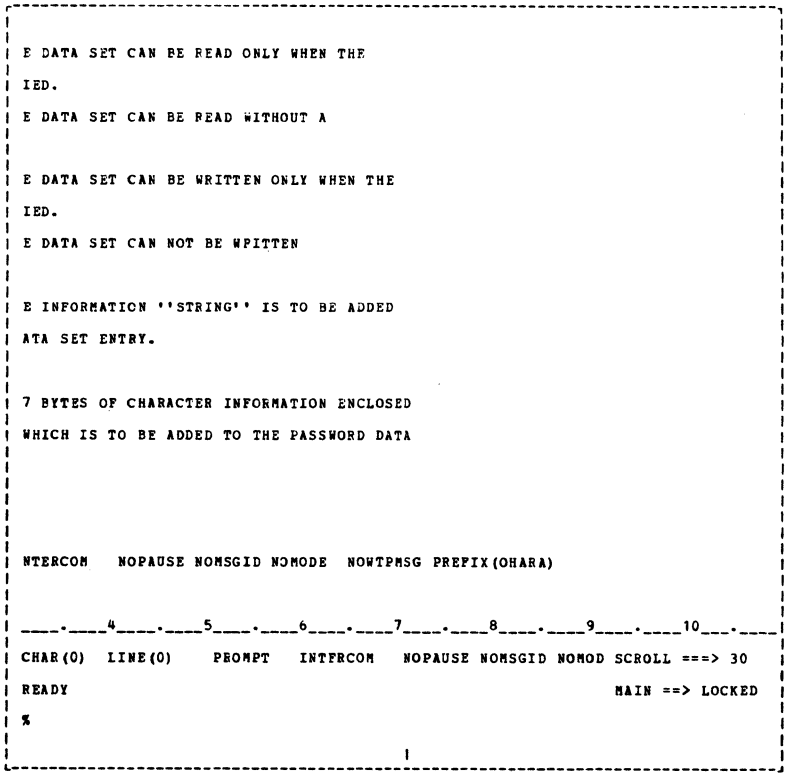


Figure 1.14 - PF 11 Has Been Pressed

Now press:

[PF12]

to unlock the MAIN window, so that when you enter the next TSO command the window will move to display the output from the command. Since the screen is up to date and TSO has generated no more output, the data displayed in the MAIN window will not change. Notice, however, that "UNLOCKED" is now displayed to the below the scroll amount field.

Entering Multiple Lines of Input to TSO

The TSOOUT stream was introduced earlier in this primer. Now another stream is introduced: the TSOIN stream. The TSOOUT stream

contains the complete record of your session: your input to TSO and TSO's output to you. The TSOIN stream is a subset of the TSOOUT stream: it contains just your input to TSO.

The TSOIN stream serves as an input buffer to TSO. When you type a command on the screen and press the ENTER key or a PF key, the Session Manager reads the screen and places the data in the TSOIN stream. When TSO requests a line of data from the terminal, the Session Manager gives it a line from the TSOIN stream. At the same time, this line is also copied to the TSOOUT stream to make that stream a complete record of your TSO session. This is shown in Figure 1.15.

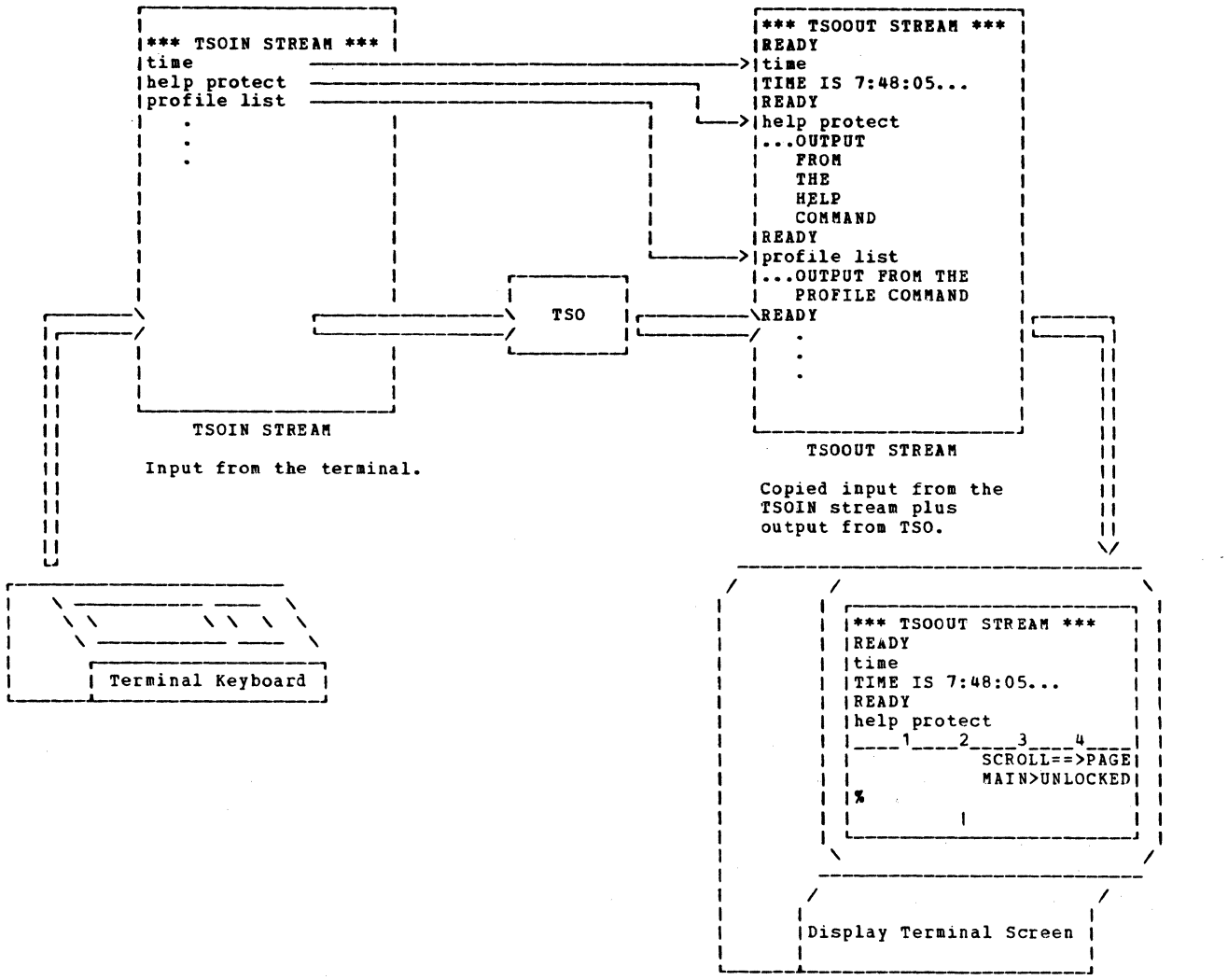


Figure 1.15 - The TSOIN and TSOOUT Streams

Until now, the CURRENT window has always viewed the TSOOUT

stream. You can change this by using the PF 6 key. Press:

[PF6]

and window CURRENT will now view the TSOIN stream. Your screen should look like Figure 1.16. Notice that the CURRENT window now displays the last two TSO commands you entered.

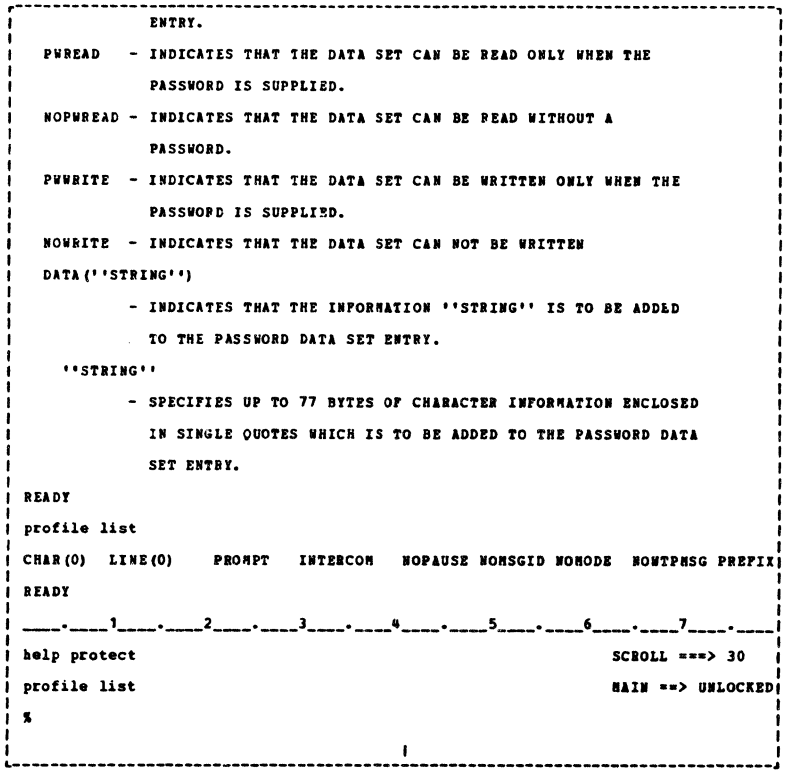


Figure 1.16 - PF 6 Has Been Pressed

You can enter more than one TSO command at a time, simply by entering them on different lines of the screen. For example, move the cursor up one line and enter the following TSO commands as a group by pressing the NEW LINE key (shown as "[<—┘]") after the first (instead of the ENTER key):

listalc [<—┘]

listcat [ENTER]

Notice the two commands you entered are displayed in the CURRENT window, and that the LISTALC command is highlighted. When it finishes executing, notice the LISTCAT command is highlighted, and when it has finished executing, they will both be displayed at normal intensity. You may need to press [ENTER] once or twice to allow the Session Manager to update the screen. Your screen

should look like Figure 1.17.

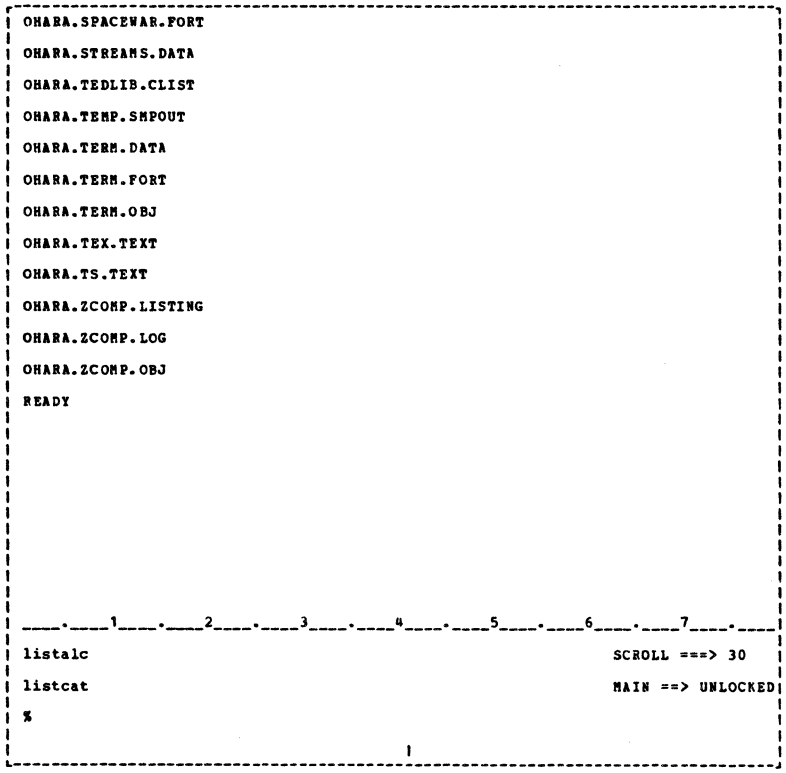


Figure 1.17 - The LISTALC and LISTCAT Commands Have Been Entered

When you enter multiple commands or other input as a group (as was just done), the order in which you enter the commands is <u>not</u> the order of execution. The Session Manager does not know in which order you typed the commands. It is the placement of the data you type on the screen that determines the order of execution. The display screen is read from <u>top</u> <u>to</u> <u>bottom</u> when you press the ENTER key or one of the PF keys. For example, move the cursor up a few lines into the MAIN window and type:

time

and then above it enter:

listbc [ENTER]

You'll notice that the LISTBC command was executed first, even though the TIME command was entered first. This is shown in Figure 1.18.

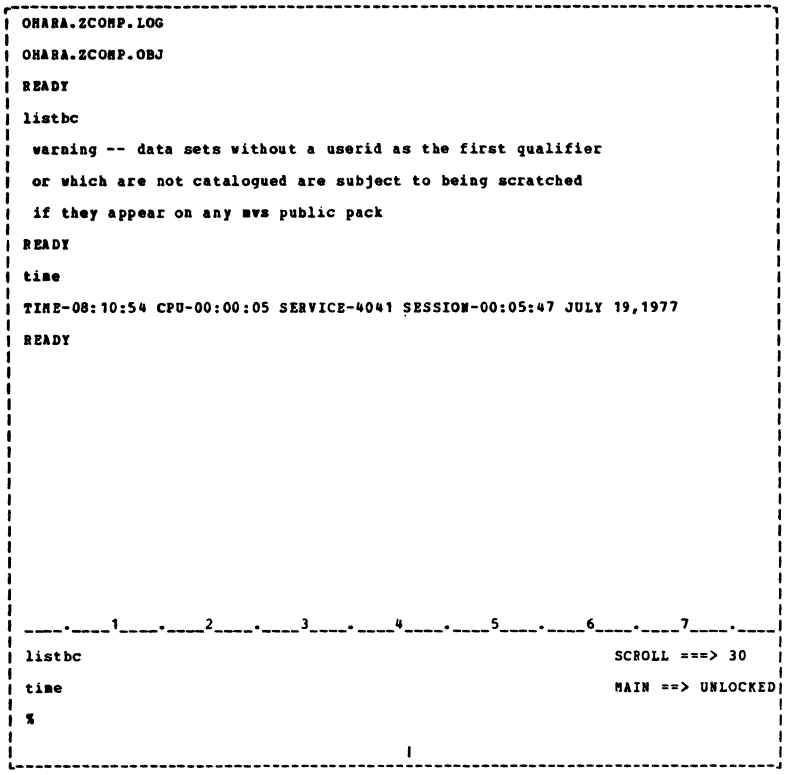


Figure 1.18 - The LISTBC and TIME Commands Have Been Entered

You can return the CURRENT window to viewing the TSOOUT stream by pressing:

[PF6]

again. PF 6 allows you to switch the CURRENT window between viewing the TSOIN and TSOOUT streams.

Using TSO EDIT with the Session Manager

TSO EDIT and the EDIT subcommands act somewhat differently under the Session Manager than under normal TSO. The Session Manager, by allowing you to control what is displayed on the screen, makes EDIT easier to use. It also makes some functions (such as automatic line numbering in INPUT mode) unneeded. To edit a data set using TSO EDIT enter:

edit sample [ENTER]

and your screen should look like Figure 1.19. Since the data set type was not entered with the EDIT command you have been prompted

by EDIT to enter the type of data set to be edited. Notice that the prompting message is displayed both in the CURRENT window and in the MAIN window. If the MAIN window had been scrolled back to some previous data (and locked there), we would not have seen the prompting message. This is why the CURRENT window normally views the TSOOUT stream and is not locked: to ensure that you will always see any prompting or error messages, even when the MAIN window is locked.

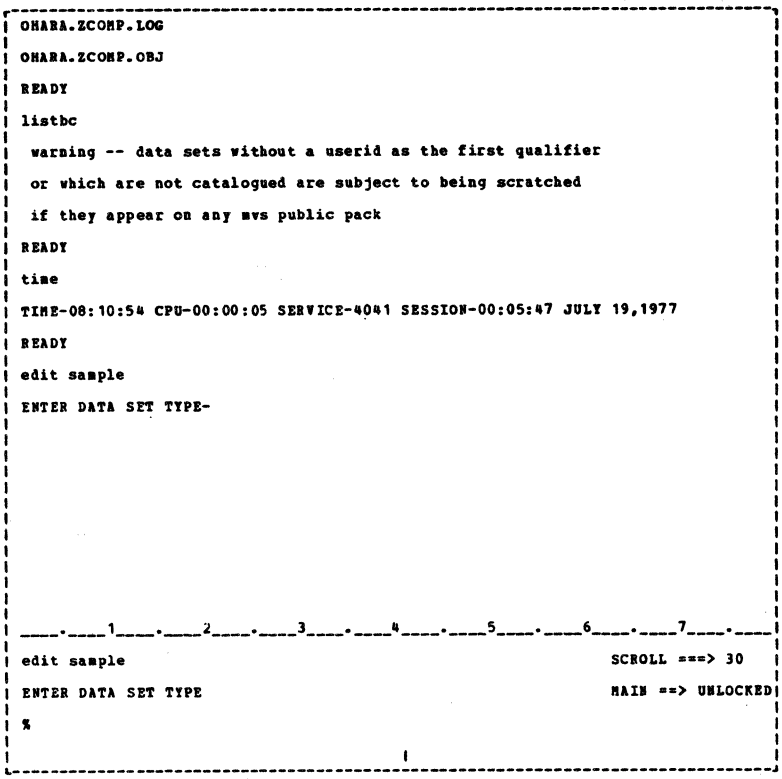


Figure 1.19 - The EDIT Command Has Been Entered

The SAMPLE data set is going to be a Command Procedure (CLIST), so enter:

clist [ENTER]

to respond to the prompting message. Your screen should look like Figure 1.20.

Notice that there is no automatic line numbering. EDIT is in INPUT mode but has not prompted you with a line number, as it would at a typewriter terminal or a display terminal without the Session Manager. Line numbers are not needed because you may enter multiple lines of input at one time.

Note: If you wish to be prompted with a line number before enter-

ing each line of input, specify the "R" operand of the INPUT subcommand of EDIT. The line numbers and your input will be displayed on separate lines in the TSOOUT stream.

```
OHARA.ZCOMP.LOG
OHARA.ZCOMP.OBJ
READY
listbc
 warning -- data sets without a userid as the first qualifier
 or which are not catalogued are subject to being scratched
 if they appear on any mvs public pack
READY
time
TIME-08:10:54 CPU-00:00:05 SERVICE-4041 SESSION-00:05:47 JULY 19,1977
READY
edit sample
ENTER DATA SET TYPE-
clist
DATA SET OR MEMBER NOT FOUND, ASSUMED TO BE NEW
INPUT




____.____1____.____2____.____3____.____4____.____5____.____6____.____7____.____
DATA SET OR MEMBER NOT FOUND, ASSUMED TO BE NEW          SCROLL ===> 30
INPUT                                                    MAIN ==> UNLOCKED
%
                                    I
```

Figure 1.20 - In EDIT INPUT Mode

Move the cursor up onto the MAIN window, using the Cursor Positioning keys, and enter the following lines of data:

```
write *** this is the sample clist ***        [<-┘]

listcat                                       [<-┘]

time                                          [ENTER]
```

Use the NEW LINE key ("[<-┘]") to move the cursor to the next row after each entry and the ENTER key to send the lines to TSO. Your screen should now look like Figure 1.21.

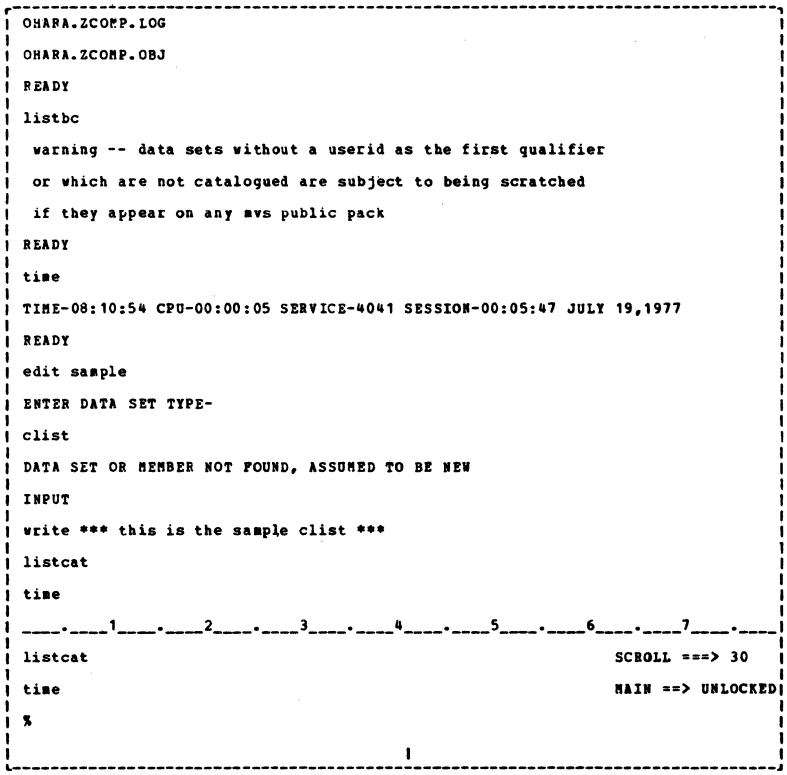


Figure 1.21 - After Entering Three Lines of Input

To end EDIT INPUT mode at a typewriter terminal or at a display terminal without the Session Manager you press the Carriage Return or the ENTER key without entering any data. This sends a "null line" (a line of zero length) to TSO and signals TSO EDIT to end INPUT mode. But, as was noted before, pressing the ENTER key alone at a Session Manager-supported terminal sends nothing to TSO. To send a null line to TSO, you must press the ERASE EOF key (shown as "[ERASE EOF]") followed by the ENTER key:

[ERASE EOF] [ENTER]

EDIT will respond with the mode message, "EDIT". To display the data set, enter:

list [ENTER]

Your screen should now look like Figure 1.22.

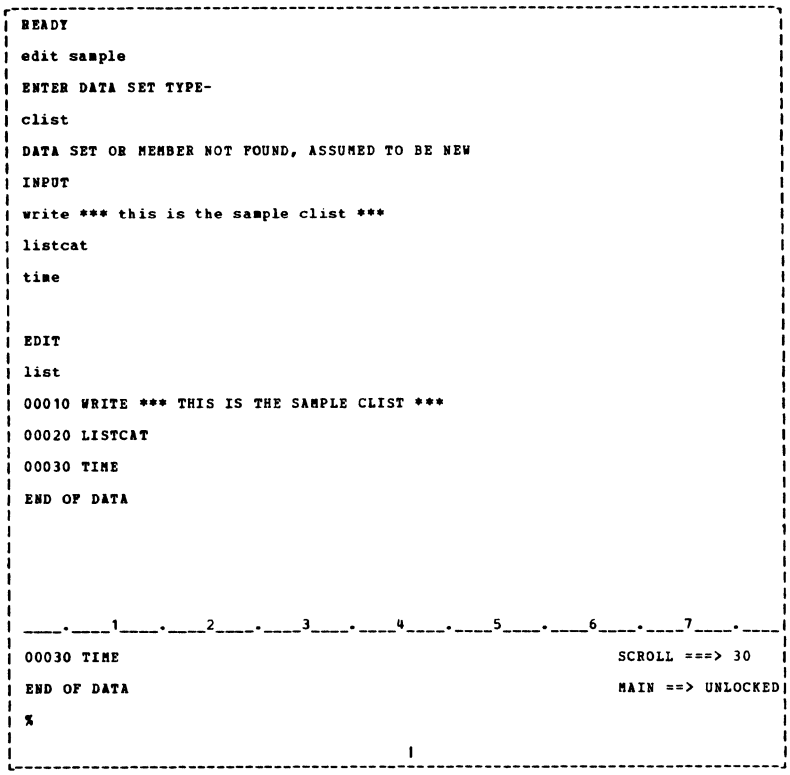


Figure 1.22 - The LIST Subcommand Has Been Entered

With the Session Manager it is possible to move the cursor into the MAIN window and directly modify the lines displayed as output from the LIST subcommand. When you press [ENTER], these new lines will be interpreted as input to EDIT and the changes will be recorded. For example, to change line 10 to read:

WRITE *** BEGIN EXECUTION OF THE SAMPLE CLIST ***

move the cursor up into the MAIN window and retype line 10 using the 3270 editing keys. Press:

[ENTER]

and line 10 will be changed.

You can also add or insert lines by altering the line numbers at the beginning of the line. To add this line:

WRITE *** END EXECUTION OF THE SAMPLE CLIST ***

at the end of the data set, you can make use of the line you just entered. Move the cursor up just one line and reuse the displayed line 10 in the CURRENT window. By changing the 10 to a 40 and "BEGIN" to "END", you will have created the new line. Now use the NEW LINE key to move the cursor down to the next row and en-

ter:

list [ENTER]

to list the data set again and see the changes in context. Your screen should look like Figure 1.23.

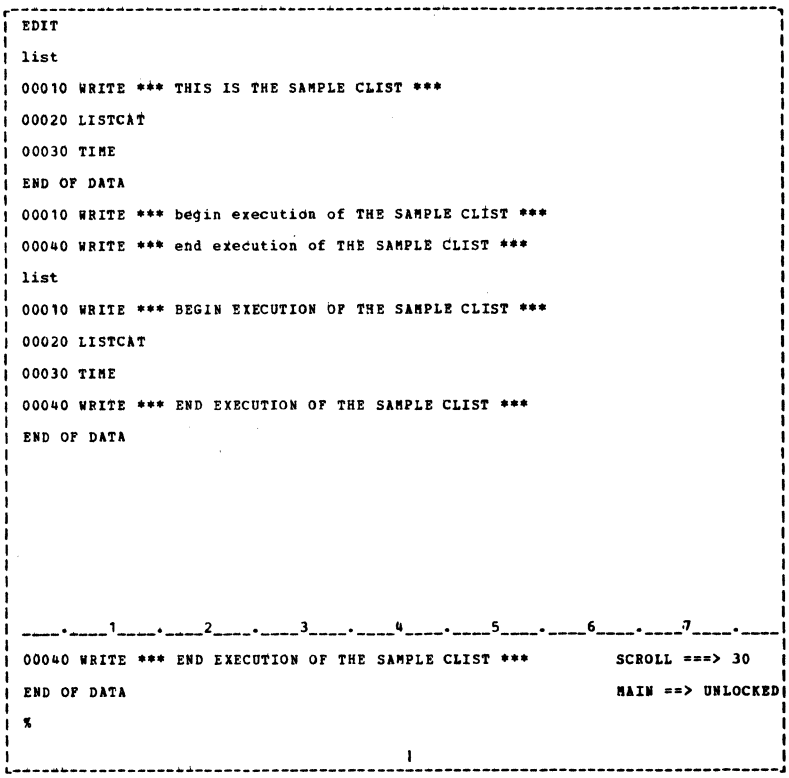


Figure 1.23 - Lines 10 and 40 Have Been Changed

You can delete lines in a similar fashion. To remove line 30, move the cursor up into the MAIN window and position it directly after "00030". Press:

[ERASE EOF]

and all the characters on that line after the line number are removed. Press:

[ENTER]

and just the line number will be sent to TSO. EDIT will interpret this as a command to delete that line.

Note: In the SAMPLE.CLIST, all of the lines were shorter than the width of the screen. If a line in your data set is longer than the screen and you modify it on the screen, the undis-

played (truncated) data is lost. For those lines you must use the EDIT subcommand CHANGE. You can use the TSO line continuation characters ("+" and "-") to avoid such long lines. The EDIT subcommands and the use of line continuation characters are described in "TSO Command Language Reference", GC28-0646.

To list the data set and then save it, move the cursor to the first line of the CURRENT window and enter:

```
list     [<-┘]

end save    [ENTER]
```

Your screen should look like Figure 1.24.

```
00020 LISTCAT
00030 TIME
00040 WRITE *** END EXECUTION OF THE SAMPLE CLIST ***
END OF DATA
00030
list
00010 WRITE *** BEGIN EXECUTION OF THE SAMPLE CLIST ***
00020 LISTCAT
00040 WRITE *** END EXECUTION OF THE SAMPLE CLIST ***
END OF DATA
end save
READY
____.____1____.____2____.____3____.____4____.____5____.____6____.____7____.____
end save                                          SCROLL ===> 30
READY                                             MAIN ==> UNLOCKED
%
```

Figure 1.24 - The END Subcommand Has Been Entered

Using Displayed Data to Form New Input

To execute SAMPLE.CLIST enter:

exec sample [ENTER]

When its execution has completed, the READY message will appear. Your screen should look similar to Figure 1.25. The output from your CLIST will be different, of course, because the data sets in your catalog will not be the same as those displayed here. Depending on the response time at your installation, you may need to press the ENTER key once or twice to tell the Session Manager to update the screen so that all of the output will be displayed.

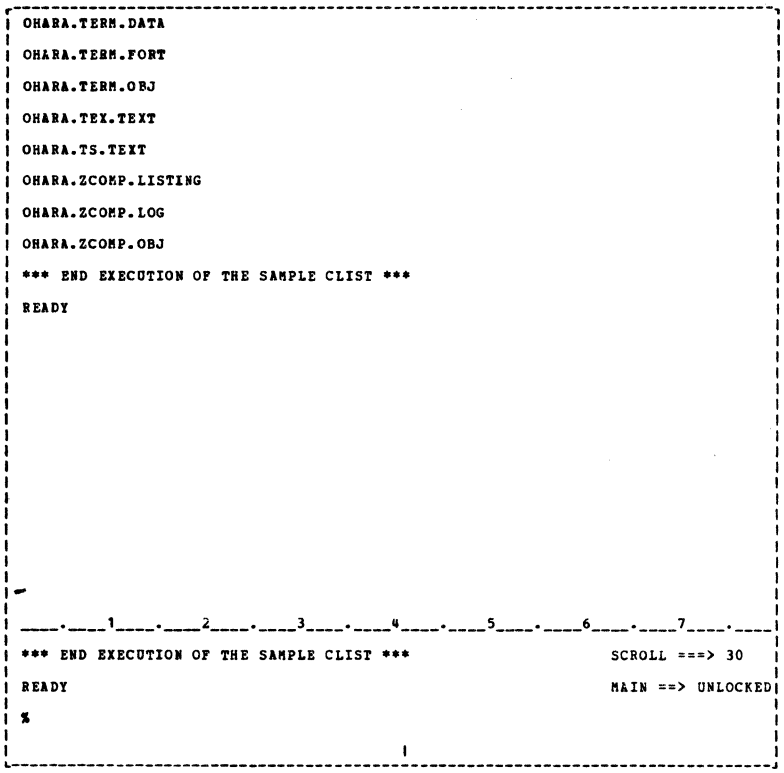


Figure 1.25 - Execution of SAMPLE.CLIST

Earlier in this primer, you used data displayed on the screen (the listing of the SAMPLE.CLIST data set) to form new input to EDIT (to add, modify, and delete lines). In the same manner, you can use the output from the LISTCAT command in the SAMPLE.CLIST to form new command input, thus saving time and avoiding typing errors. Press:

[PF7]

to scroll backward through your catalog listing and display the listing of the SAMPLE.CLIST data set. You may wish to reset the scroll amount to a new value for scrolling over the listing of datasets. To set the scroll amount to half a page, enter:

half [PF2]

Pressing PF 7 will now cause the window to scroll back one half page each time it is pressed.

Note: You may need to press PF 7 more than once or not at all (if SAMPLE.CLIST is already displayed), depending on the number of data sets in your catalog.

You can easily rename the data set SAMPLE.CLIST to one called SAMPLE1.CLIST by moving the cursor up to where the data set name is displayed in the window. By using the terminal editing keys, you can create the RENAME command as shown below and in Figure 1.26, (the seventh line above the dashed line).

rename SAMPLE.CLIST sample1.clist

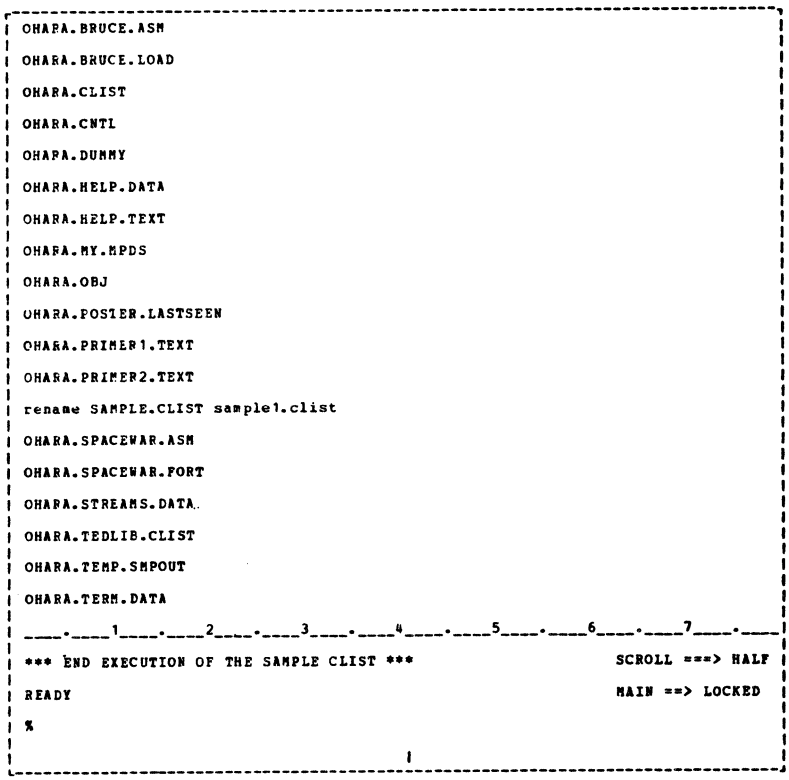


Figure 1.26 - Renaming SAMPLE.CLIST

If you press the ENTER key, the RENAME command will be executed. But because you have scrolled back (by using PF 7), the window is locked ("LOCKED" is displayed below the scroll amount field) and you must press PF 12 to unlock the window. To avoid pressing the ENTER key and then PF 12, you can simply press:

[PF12]

for this performs the function of the ENTER key and then the function of PF 12 (moving the MAIN window to display the newest data in the TSOOUT stream and unlocking it there). The result is shown in Figure 1.27. Notice that "UNLOCKED" is now displayed below the scroll amount field.

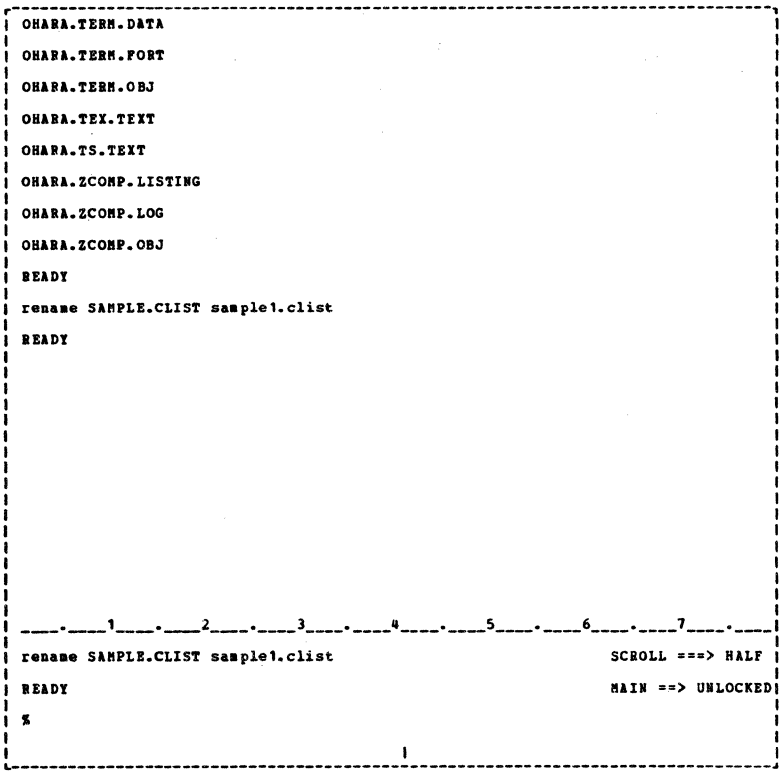


Figure 1.27 - PF 12 Has Been Pressed

All of the Program Function keys except PF 2 and PF 5 operate in a similar manner: if you type data on the screen before pressing the function key, the data will be sent to TSO (just as if you had pressed the ENTER key) before the action of the function key is performed. (The data you type on the screen before pressing PF 2 sets the scroll amount for the scrolling PF keys; for PF 5 the data you type is used as a search argument.)

Entering Passwords with the Session Manager

TSO provides a data set security facility which allows you to protect your data sets from unauthorized use. To accomplish this you can assign one or more passwords to your data set. Then, in order to access the data set, the valid password must be specified. To protect your data set SAMPLE1.CLIST by giving it a

password of "josentme", enter:

```
protect sample1.clist add(josentme)     [ENTER]
```

"JOSENTME" is now the password for the data set. If you try to access the data set or want to change the password in the future, you will be prompted by TSO to enter the password. The Session Manager will recognize the situation and, to maintain security, will not display the password in the stream. For example, to remove the password assigned to SAMPLE.CLIST, enter:

```
protect sample1.clist delete     [ENTER]
```

and you will be prompted for the password. You can reuse the PROTECT command you just entered, now displayed in the CURRENT window if you wish, by changing "add(josentme)" to "delete" before pressing the ENTER key. Your screen should now look like Figure 1.28. Notice the TSO message prompting you to enter the password.

```
OHARA.TERM.DATA
OHARA.TERM.FORT
OHARA.TERM.OBJ
OHARA.TEX.TEXT
OHARA.TS.TEXT
OHARA.ZCOMP.LISTING
OHARA.ZCOMP.LOG
OHARA.ZCOMP.OBJ
READY
rename SAMPLE.CLIST sample1.clist
READY
protect sample1.clist add(josentme)
READY
protect sample1.clist delete
ENTER CURRENT PASSWORD -




____.____1____.____2____.____3____.____4____.____5____.____6____.____7____.____
protect sample1.clist delete                          SCROLL ===> HALF
ENTER CURRENT PASSWORD                                MAIN ==> UNLOCKED
%
```

Figure 1.28 - The PROTECT Command Has Been Entered

You now have two ways to enter the password:

1. You may leave the cursor where it is and type the password, in which case the password will be displayed as you type it.

Enter:

josentme [ENTER]

Once you press the ENTER key, the data you typed will disappear and will not be displayed in the stream. Your screen should then look like Figure 1.29.

2. If you do not want the password displayed even as you type it (perhaps someone is looking over your shoulder), move the cursor to the area in the lower right hand corner of the screen (just to the right of the vertical bar) by using the TAB FORWARD key (shown as "[->|]"). Press:

[->|]

and your cursor will move to the password entry area. Now enter:

josentme [ENTER]

Notice that the password is not displayed as you type it. Your screen should look like Figure 1.29.

```
OHARA.TERM.DATA
OHARA.TERM.FORT
OHARA.TERM.OBJ
OHARA.TEX.TEXT
OHARA.TS.TEXT
OHARA.ZCOMP.LISTING
OHARA.ZCOMP.LOG
OHARA.ZCOMP.OBJ
READY
rename SAMPLE.CLIST sample1.clist
READY
protect sample1.clist add(josentme)
READY
protect sample1.clist delete
ENTER CURRENT PASSWORD -

READY



____.____1____.____2____.____3____.____4____.____5____.____6____.____7____.____
                                                SCROLL ===> HALF
READY                                           MAIN ==> UNLOCKED
%
```

Figure 1.29 - The Password "josentme" Has Been Entered

Obtaining Hardcopy of the Screen or Streams

The Session Manager provides the SMCOPY command by which you can produce a listing of the data in the TSOOUT stream. Using this command, you can copy the TSOOUT stream (the log of your TSO session) to the system printer. To make a copy of the TSOOUT stream, enter:

smcopy format caps [ENTER]

The CAPS and FORMAT keywords are optional. If you enter the CAPS operand, lowercase characters in the stream will be translated to uppercase before they are printed. This is necessary if the printer does not print lowercase characters; otherwise, lowercase characters are unprintable. If you enter the FORMAT operand, the highlighted lines in the TSOOUT stream will be printed darker on the printed copy.

Note: A complete description of the SMCOPY command is provided in "The Session Manager Command Language".

Sometimes it is useful to have a record of the display screen as it looks at any given moment. Pressing:

[PF4]

causes a "snapshot" of the display screen to be placed in another Session Manager stream (the "EXTRA1" stream). This stream has a capacity of at least six snapshots; when it fills up, the earlier snapshots are overlayed and lost. To print the stream containing the snapshots at a system printer, press:

[PF1]

The EXTRA1 stream is printed, then the contents are erased to make way for new snapshots of the screen image. A sample snapshot taken by pressing PF 4 and PF 1 is shown in the SNAPSHOT command, in "Session Control Commands".

Note: An SMCOPY command is generated when you press PF 1 to copy the image to the system printer. Thus you can not press PF 1 while you are in subcommand mode under another TSO command such as EDIT or TEST. You may, however, press PF 4 at any time to record the screen image, and press PF 1 later when you are out of subcommand mode.

Receiving Messages From Other Users

At a display terminal without the Session Manager, messages sent to you by other TSO users, the operator, or from background jobs are displayed on the screen, disrupting your session. Also, once you clear the screen, the message is lost.

The Session Manager eliminates these problems by intercepting the message and placing it in the TSOOUT stream, highlighted. The audible alarm (if present) will sound to alert you that you have received a message. You can press [PF12] to move the main window to the newest data in the TSOOUT stream to view the message.

Using Full Screen Programs

The Session Manager has been designed so as to not interfere with other TSO full screen programs your installation may have installed. An example of such a program is SPF, the Structured Programming Facility, program number 5740-XT2. This is a Program Product available from IBM for a license fee. If your installation does not have SPF, skip this section of the primer. This primer assumes you are familiar with using SPF. If you are not, read the "TSO 3270 Structured Programming Facility (SPF) Reference Manual", SH20-1730.

Using SPF while under the Session Manager is similar to using it with normal TSO, except when you get to Option 4 or 6 of SPF. These options allow you to interface with the TSO Program Prompters (option 4) or selected TSO commands (option 6). The use of Option 6 with the Session Manager will be illustrated here, but it also applies to SPF Option 4. To invoke SPF, enter:

spf [ENTER]

When the SPF Primary Menu is displayed enter:

6 [ENTER]

The SPF Option 6 Menu will now be displayed. Under Option 6 of SPF enter the TSO command:

listds sample1.clist [ENTER]

Your screen should now look like Figure 1.30. (You may need to press the ENTER key to have the screen updated.) Notice that the Session Manager has once again taken control of the display screen. Instead of a TSO "READY" message after the output of the LISTDS command there is a Session Manager message prompting you to enter a null line. Until you do this (by using the ERASE EOF and ENTER keys), you may use the Session Manager scrolling func-

tion keys to review any of the output of your TSO session including the TSO command you just entered. You may not, however, enter new TSO commands.

```
protect sample1.clist add(josentme)
READY
protect sample1.clist delete
ENTER CURRENT PASSWORD -


READY
smcopy format caps
READY
spf
OHARA.SAMPLE1.CLIST
--RECFM-LRECL-BLKSIZE-DSORG
  VB    255   3120    PS
--VOLUMES--
  VSTR07
ADF041A ENTER A NULL LINE TO RETURN TO FULL SCREEN PROGRAM









____.____1____.____2____.____3____.____4____.____5____.____6____.____7____.____
  VSTR07                                          SCROLL ===> HALF
ADF041A ENTER A NULL LINE TO RETURN TO FULL SCREEN PROGRAM    MAIN ==> UNLOCKED
%
```

Figure 1.30 - The LISTDS Command Entered Under SPF Option 6

When you wish to return to SPF, generate a null line by pressing:

[ERASE EOF] [ENTER]

You will return to SPF's Option 6 menu from which you may enter another TSO command or move to other menus of SPF. When you leave SPF, your screen should look like Figure 1.31.

Note: If you submitted a background job with the SPF Exit Menu, the Session Manager will again prompt you to enter a null line to continue your TSO session.

Notice that none of the work you did under SPF has been recorded in the Session Manager stream. SPF provides separate journaling facilities. If however, you received any messages from other users, background jobs, or the operator while you were in SPF, then a copy of each message is placed in the TSOOUT stream.

```
protect sample1.clist add(josentme)
READY
protect sample1.clist delete
ENTER CURRENT PASSWORD -

READY
smcopy format caps
READY
spf
OHARA.SAMPLE1.CLIST
--RECFM-LRECL-BLKSIZE-DSORG
  VB     255   3120    PS
--VOLUMES--
  VSTR07
ADF041A ENTER A NULL LINE TO RETURN TO FULL SCREEN PROGRAM

READY



____.____1____.____2____.____3____.____4____.____5____.____6____.____7____.____
                                                        SCROLL ===> HALF
READY                                                   MAIN ==> UNLOCKED
%
```

Figure 1.31 - After Exiting From SPF

2. USING THE SESSION MANAGER COMMANDS

Introduction

This section will introduce you to using the Session Manager commands to control and modify your display terminal environment. It is assumed that you have read the Session Manager primer and are now familiar with the use of the IBM-supplied default screen layout and Program Function key definitions, and that you have some knowledge of TSO. When a Session Manager command is introduced it is suggested that you turn to "The Session Manager Command Language" and read the description of the command and its operands.

Two more Session Manager streams are introduced at this time. Just as there is an input and output stream for TSO (TSOIN and TSOOUT), there is an input and output stream for the Session Manager, named respectively, SMIN and SMOUT. When you enter a Session Manager command, it is placed in the SMIN stream. It is also copied to the SMOUT stream, just as TSO commands are copied from the TSOIN stream to the TSOOUT stream. If there should be an error in a Session Manager command, an error message is placed in the SMOUT stream, highlighted. This is shown in Figure 2.1.

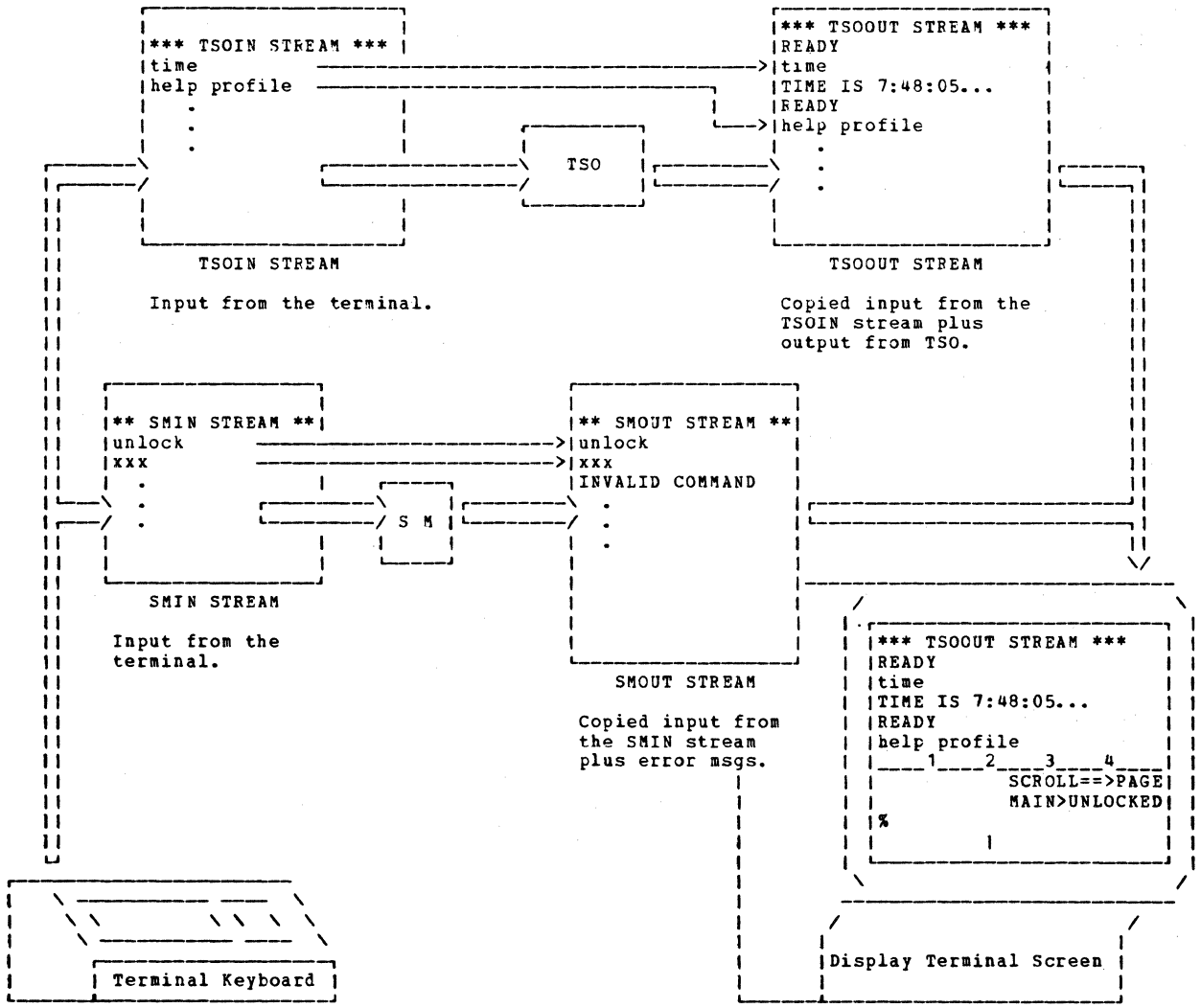


Figure 2.1 - TSO and Session Manager Input and Output Streams

As can be seen in Figure 2.1, the Session Manager operates in parallel with TSO. This allows you to enter and execute Session Manager commands while TSO commands are executing. Thus while waiting for a long-running TSO command to execute (a compilation, for example), you can issue any Session Manager commands you wish. You can, for example, scroll a window over the TSOOUT stream to review the data of your session, or perhaps redefine a Program Function key for use when the compilation finishes.

Entering Session Manager Commands

As described in the primer, you can press the CLEAR key and then enter Session Manager commands on the line following the message, but this prevents you from viewing the data displayed on the screen at the same time. Pressing PF 3 sets up the screen to allow you to enter Session Manager commands without using the CLEAR key. To begin Session Manager command entry, press:

[PF3]

This does several things:

1. Any data you type before pressing PF 3 is sent to the SMIN stream and interpreted as Session Manager command(s). Note that in this case no data was typed before PF 3 was pressed.

2. PF key 3 now serves as an ENTER Key for Session Manager commands. Typing data in an unprotected window and pressing PF 3 will send the data to the SMIN stream, to be interpreted as a Session Manager command.

3. The CURRENT window now views the SMOUT stream (containing Session Manager commands you have entered and any error messages resulting from them).

   Note: The stream is initially blank. When you enter a Session Manager command or press one of the PF keys, you will see the Session Manager commands in the SMOUT stream.

4. Data typed in the CURRENT window will be sent to the SMIN stream and interpreted as a Session Manager command when, subsequently, the ENTER key or one of the PF keys is pressed.

5. PF 6 will restore window CURRENT and PF 3 to their previous meanings, as explained in the Session Manager primer.

To illustrate the action of the SMOUT stream, enter an invalid Session Manager command:

xxx [PF3]

Your screen should now look like Figure 2.2. The audible alarm (if your terminal is so equipped) should have sounded. Notice the error message displayed in the CURRENT window.

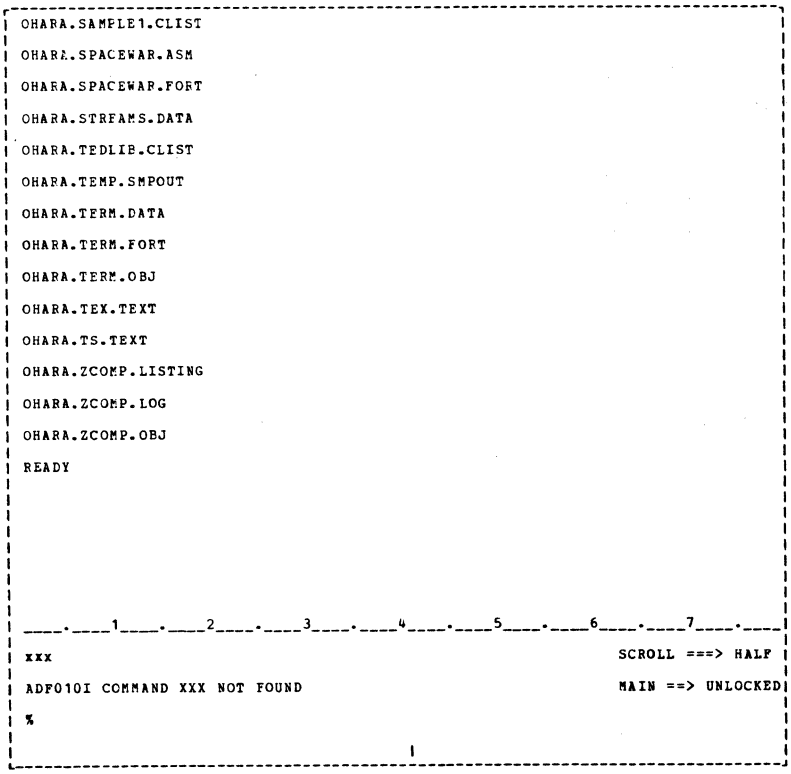


Figure 2.2 - An Invalid Session Manager Command Has Been Entered

Viewing Different Streams

Often it is useful to have the MAIN window view streams other than TSOOUT. For example, you can review the TSO commands you have issued during your session by viewing the TSOIN stream through the MAIN window. To do this, enter:

```
change.window main view(tsoin)    [PF3]
```

The MAIN window will now display data in the TSOIN stream. It is unlocked, and is positioned over the newest data in the stream, so it displays the last 19 TSO commands you issued during this session. As described in "A Session Manager Primer", you can use the scrolling PF keys to move the MAIN window over the TSOIN stream and review commands you entered earlier in the session.

Viewing the TSOIN stream through the MAIN window is useful to execute many commands at one time. For example, if you wanted to compile several programs, you can enter all the commands to do the compilations at one time and then monitor the execution of the commands: the command currently executing will be highlighted

in the TSOIN stream. This was illustrated earlier in the Primer.

To view the data in the SMOUT stream (containing Session Manager commands and error messages), enter:

```
change.window main view(smout)    [PF3]
```

The MAIN window will now display data in the SMOUT stream, and is unlocked (even if it was locked when you issued the CHANGE.WINDOW command). You will see copies of the Session Manager commands you have issued either directly or through pressing PF keys, and also any error messages that have been issued by the Session Manager. The last line in the stream will be the copy of the command you just entered.

Note: The "LOCKED"/"UNLOCKED" indicator is only accurate when you use the default PF keys to scroll. Entering Session Manager commands (such as CHANGE.WINDOW) may unlock or lock the MAIN window without updating this field. Once you press one of the scrolling PF keys or PF 12 (to unlock the MAIN window) the field will be updated and again reflect the true status of the MAIN window.

The other Session Manager stream names and attributes can be displayed by the QUERY command. To do this, enter:

```
query.stream smout    [PF3]
```

The output of the QUERY command (which should now be visible in the MAIN window) lists each of the Session Manager's streams, their size, how much space has been used, and attributes of the stream. The QUERY command and its output is described in "The Session Manager Command Language".

Defining Program Function Keys

The PF Keys that were introduced in the primer do not have permanently defined values. You can use the CHANGE.PFK command to modify the meanings of each of the PF keys to suit your particular needs. To display the current definitions of the PF keys, enter:

```
query.pfks smout    [PF3]
```

The MAIN window should now display a table containing an entry for each PF key. You may need to use the scrolling PF keys to view the entire output.

As can be seen from the output of the QUERY command, each PF key is defined as a string of characters to be placed in a Session Manager stream. In the IBM-supplied default screen layout, all of

the PF keys are defined as placing their text strings in the SMIN stream. Thus, each PF key is in effect a collection of Session Manager commands.

If you define a PF key as a character string to be sent to the TSOIN stream, when pressed it will be interpreted as one or more TSO commands or input to an application program.

To illustrate how the PF keys work, enter:

```
change.pfk 1 'time' tsoin    [PF3]
```

This changes PF 1 to place the characters "time" in the TSOIN stream, where they will be (sucessfully) interpreted as the TSO TIME command. To make the MAIN window again view the TSOOUT stream (so the execution of the TIME command can be observed), enter:

```
change.window main view(tsoout)    [PF3]
```

Press:

```
[PF1]
```

and the TIME command will be executed.

PF keys can also accept input when they are pressed. When you define the text of the PF key (in the CHANGE.PFK command) you can specify symbolic arguments in the text. When you enter data on the screen and then press a PF key so defined, the input will be placed in the text string, substituting the symbolic arguments in the PF key definition text string. Using this function, you can, for example, define a PF key to issue the LISTDS command, accepting a data set name as input. Enter:

```
change.pfk 1 'listds ''&1.'' members' tsoin substitute    [PF3]
```

Notice the use of doubled quotes (needed to produce single quotes when the PF key is pressed) and the period after "&1" indicating the substituted value is to be concatenated with the rest of the PF definition. To display data sets in your catalog, enter:

```
listcat    [ENTER]
```

To list the attributes of the data set "SAMPLE1.CLIST", scroll the MAIN window until the data set name is visible (by using the scrolling PF keys). Then use the cursor movement keys to move the cursor up to the line where the data set name is displayed in the MAIN window. Insert a space in front of the data set name (this will modify the line and cause it to be sent to the computer when you press a PF key), and press:

```
[PF1]
```

then (to unlock the MAIN window) press:

[PF12]

Your screen should look like Figure 2.3. You might want to try this PF key on other data sets in your catalog.

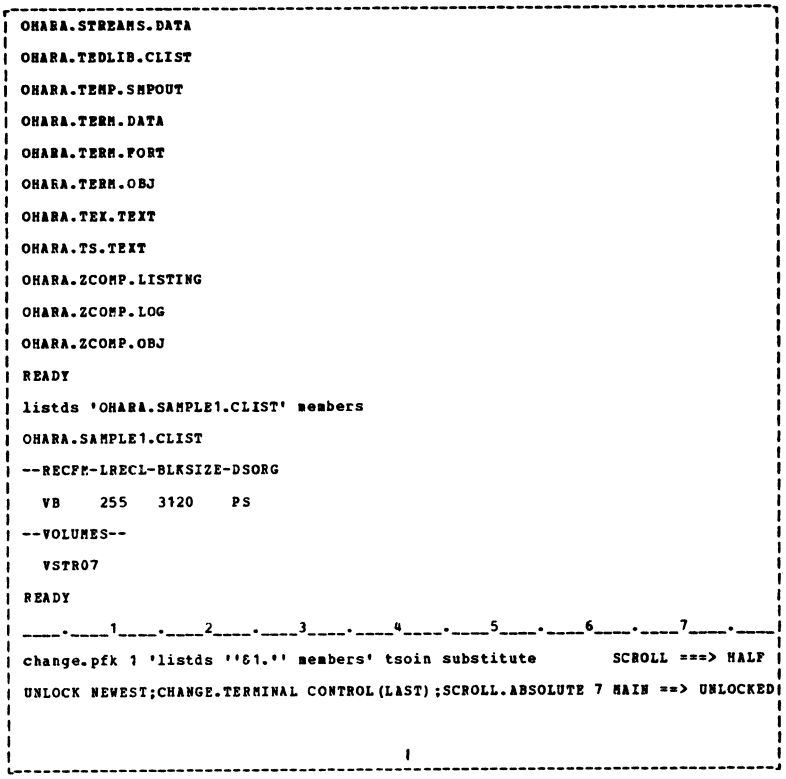


Figure 2.3 - Executing the LISTDS Command by Pressing PF 1

Notice the LISTDS command has now been executed. (You will need to press PF 12 to unlock the MAIN window if you had locked it by pressing one of the scrolling PF keys.) If you had typed a space in front of more than one data set name, the LISTDS command would have been re-executed for each of those data set names. In a similar fashion, you could have executed the DELETE command or perhaps issued a compiler command to compile one or more data sets in your catalog.

If you review the definitions of the PF keys in the default screen layout you will observe how this facility is used in PF key 2 to allow you to set the scroll amount value, in PF 3 to allow you to enter Session Manager commands, and in PF 5 to enter the search argument for the Session Manager FIND command. Notice also that a PF key can use the CHANGE.PFK command to redefine itself or other PF keys.

Controlling the Terminal Keyboard

The CONTROL operand of the CHANGE.TERMINAL command allows you to control when your keyboard is unlocked and input is allowed. The only time you may enter commands or press the 'ATTN' key is when the keyboard is unlocked. However, the only time the Session Manager can write to the terminal and thus update the display screen is when the keyboard is locked. In standard TSO without the Session Manager, the keyboard is always locked while a command or program is executing. Thus output from the command or program can always be written to the display screen. Unfortunately, this prevents you from entering new commands while previous ones are executing.

Note: With TSO/VTAM you may press the RESET key when the keyboard is locked to unlock it and allow entry of new commands.

The CONTROL operand allows you to specify the number of seconds the keyboard should be locked while the TSO command or program is executing, or while the Session Manager is updating the display screen. If you set CONTROL to a very high number, the Session Manager will leave the keyboard locked while TSO commands are executing, and will be able to update the display screen if the TSO commands produce new output. If you set CONTROL to a very low number (zero, for example) the keyboard will never lock while a TSO command is executing. You will always be able to enter new commands. However, the Session Manager will then only be able to update the screen after you have pressed the ENTER key or a PF Key. It then becomes necessary to strike the ENTER key every time you want new output from a command or a program displayed.

Setting CONTROL to a small number (10 to 20) is a compromise between the two extremes. This allows the keyboard to be locked long enough for most TSO commands to execute and produce output, yet not so long that a long running command or program delays entry of new input.

In the IBM-supplied default screen layout, CONTROL is set to 15, and the scrolling PF keys reset the CONTROL timer to 0 in addition to scrolling the MAIN window. PF 12 resets the CONTROL timer to the last non-zero value it had (still 15 unless you have entered the CHANGE.TERMINAL command to change the CONTROL timer) as well as unlocking the window. This is done to allow you to scroll back and review previous data while a TSO command is executing without waiting a long time. When the window is unlocked (after pressing PF 12), the CONTROL setting of 15 seems to be a good compromise, as described above.

For example, enter:

```
change.terminal control(30)     [PF3]
```

Press:

[PF9]

to scroll the MAIN window back to the oldest data in the TSOOUT stream. Now enter:

unlock here [PF3]

Pressing PF 9 locked the window over the oldest data in the stream. The UNLOCK command unlocked the window, and now the Session Manager will move the window toward the newest data.

Note: "LOCKED" is still displayed below the scroll amount field, even though the MAIN window is now unlocked. This field is only updated to "UNLOCKED" when PF 12 is pressed.

Enter:

[ENTER]

to begin the movement of the window. Notice the screen updates every second (see the HOLD operand of the CHANGE.WINDOW command) as the Session Manager moves the MAIN window toward the newest data. After 30 seconds, the movement stops. This is because the CONTROL timer has expired.

If you press the ENTER key again, the movement will resume. It will stop again after 30 more seconds have elapsed, or when the window reaches the bottom of the stream (whichever comes first).

You can move the window directly to the bottom of the stream by pressing PF 12, or you can lock it in place by pressing one of the scrolling PF keys.

Thus, if you wish to review a great deal of data in the TSOOUT stream, you can set CONTROL to be a large number and allow the window to move toward the newest data under control of the Session Manager, without any interaction required on your part.

Modifying the Default Screen Layout

One of the features of the Session Manager is that you can modify the default screen layout given to you when you logged on to meet your specific needs. For example, you might wish to change the stream in which messages from other TSO users are placed. In the IBM-supplied Default Screen the layout, these messages are placed in the TSOOUT stream, displayed at high intensity, and the alarm is sounded. You can use the CHANGE.FUNCTION command to redirect these messages to the MESSAGE stream, rather than have them interspersed with other output in the TSOOUT stream. Then, when you receive a message, the message will be placed in the MESSAGE stream, and the alarm will sound. You can use the CHANGE.WINDOW command to then view the MESSAGE stream and read the message.

While you could issue these Session Manager commands to modify the default screen layout by typing them, it is easier to create a TSO CLIST that will execute the commands for you. Such a CLIST is shown in Figure 2.4.

```
0000100 /*********************************************************************/
0000200 /* THIS CLIST SETS UP A SESSION MANAGER PROGRAM FUNCTION KEY.        */
0000300 /* IT ASSUMES THE IBM-SUPPLIED DEFAULT SCREEN LAYOUT IS THE CURRENT  */
0000400 /* SCREEN LAYOUT WHEN IT IS INVOKED:                                 */
0000500 /* - THE "MSG" FUNCTION IS CHANGED TO SEND MESSAGES FROM OTHER TSO   */
0000600 /*   USERS TO THE "MESSAGE" STREAM, SOUNDING THE ALARM WHEN A        */
0000700 /*   MESSAGE IS RECEIVED.                                            */
0000800 /* - PF KEY 9 IS DEFINED TO MAKE THE "MAIN" WINDOW VIEW THE MESSAGE  */
0000900 /*   STREAM WHEN PRESSED. IF PRESSED AGAIN, PF 9 RETURNS THE "MAIN"  */
0001000 /*   WINDOW TO VIEWING THE STREAM IT WAS VIEWING WHEN PF 9 WAS FIRST */
0001100 /*   PRESSED.                                                        */
0001200 /* - THE EXISTING PF KEYS ARE SAVED BEFORE THE SCREEN IS MODIFIED.   */
0001300 /*********************************************************************/
0001400                                                                  /* */
0001500 SMPUT /SAVE.PFK;+
0001600        CHANGE.FUNCTION MSG OUTPUT(MESSAGE 1) ALARM(OUTPUT);+
0001700        CHANGE.PFK 9 'SAVE.PFK;SAVE.WINDOW MAIN;+
0001800                     CHANGE.WINDOW MAIN VIEW(MESSAGE);+
0001900                     CHANGE.PFK 9 ''RESTORE.PFK;RESTORE.WINDOW MAIN'' +
0002000                     SMIN' SMIN/ SMIN
```

Figure 2.4 - A TSO CLIST to Redefine PF 9

This CLIST uses the SMPUT TSO command to place Session Manager commands in the SMIN stream. It illustrates several points:

- The existing PF key definitions are first saved on the Session Manager "stack" (see the SAVE and RESTORE commands for more details). This allows you to issue the RESTORE.PFK command to change PF 9 back to its original definition if you want.

- The CHANGE.FUNCTION command is used to direct messages from other TSO users to your MESSAGE stream.

- PF 9 is defined to perform four operations:

  1. The PF key definitions (including the present definition of PF 9) are saved on the PFK stack. This allows the meaning of PF 9 to be restored when it is pressed a second time (see #4, below).

  2. The MAIN window is saved on the WINDOW stack. This allows the MAIN window's position on the stream it was viewing and its locked/unlocked status to be restored when PF 9 is pressed a second time (see #4, below).

  3. The MAIN window is changed to view the MESSAGE stream, so you can read the message from another TSO user.

4. PF 9 is now redefined to perform two operations:

   a. The MAIN window is restored to where it was when PF 9 was first pressed.

   b. The PF keys definitions are restored from the stack. This resets PF 9 to its previous definition.

- The TSO continuation character "+" was used in the "SMPUT" command. By placing several Session Manager commands (separated by semicolons) in the SMPUT command the CLIST executes noticeably faster.

  Note: Be sure the line placed in the SMIN stream (after TSO has linked the continued lines together) does not exceed 512 characters. This is the size of the Session Manager input buffer.

This CLIST makes PF 9 act as a "flip-flop" switch; pressing it once makes the MAIN window view the MESSAGE stream, pressing it again restores everything to the way it was before you pressed the PF key.

You are encouraged to create this CLIST and then execute it. When you execute it for the first time, be sure to press PF 3 first (to make the CURRENT window view the SMOUT stream). This will allow you to see the Session Manager commands your CLIST generates and any error messages that may be issued. If you wish to restore the PF keys to the definitions they had before you executed the CLIST, enter:

```
restore.pfk    [PF3]
```

To change the placement of messages back to the TSOOUT stream, enter:

```
change.function msg output(tsoout 2)    [PF3]
```

Using a CLIST to Split the Screen

Often it is useful to view two parts of the TSOOUT stream at the same time. For example, you could view the listing of a program in the top window while using TSO TEST or some other debugging facility in the bottom window. You could also view HELP information or other data in the top window while entering new commands in the bottom window. A TSO CLIST that will issue the needed Session Manager commands to split the MAIN window into two windows is shown in Figure 2.5.

```
0000100 PROC 0 LINE(10)
0000200 /*********************************************************************/
0000300 /* THIS CLIST SETS UP A "SPLIT - SCREEN" SESSION MANAGER SCREEN      */
0000400 /* LAYOUT. IT ASSUMES THE IBM - SUPPLIED DEFAULT SCREEN LAYOUT IS    */
0000500 /* THE CURRENT SCREEN LAYOUT WHEN IT IS INVOKED:                     */
0000600 /* - THE WINDOW "LINE" IS MOVED TO THE ROW SPECIFIED BY "LINE".      */
0000700 /* - THE WINDOW "CURRENT" IS EXPANDED TO FILL IN THE SPACE CREATED.  */
0000800 /* - THE SPACE WHERE WINDOW "CURRENT" FORMERLY WAS IS REPLACED BY    */
0000900 /*   THE "OLDCUR" WINDOW, WHICH IS PROTECTED.                        */
0001000 /* - THE SCREEN AND PF KEYS ARE SAVED BEFORE THE SCREEN IS MODIFIED. */
0001100 /*********************************************************************/
0001200                                                                 /* */
0001300 /* EXIT IF THE LINE OPERAND IS OUT OF RANGE                        */
0001400                                                                 /* */
0001500 IF &LINE < 2 | &LINE > 21 THEN +
0001600   EXIT
0001700                                                                 /* */
0001800 /* COMPUTE THE SIZES OF THE WINDOWS TO BE DEFINED                  */
0001900                                                                 /* */
0002000 SET &TOPS = &LINE - 1
0002100 SET &CUR = &LINE + 1
0002200 SET &CURS = 21 - &CUR
0002300                                                                 /* */
0002400 /* SAVE THE SCREEN AND THE PF KEY DEFINITIONS                      */
0002500 /* SAVE THE MAIN, LINE, AND CURRENT WINDOWS                        */
0002600 /* DELETE AND REDEFINE THE MAIN, LINE, AND CURRENT WINDOWS         */
0002700                                                                 /* */
0002800 SMPUT /SAVE.SCREEN;SAVE.PFK;+
0002900        SAVE.WINDOW CURRENT;SAVE.WINDOW LINE;SAVE.WINDOW MAIN;+
0003000        DELETE.WINDOW MAIN;DELETE.WINDOW LINE;+
0003100        DELETE.WINDOW CURRENT;+
0003200        DEFINE.WINDOW MAIN 1 1 &TOPS 80;+
0003300        RESTORE.WINDOW MAIN;+
0003400        CHANGE.WINDOW MAIN OVERLAP(1) HOLD(0)/
0003500 SMPUT /DEFINE.WINDOW LINE &LINE 1 1 80;RESTORE.WINDOW LINE;+
0003600        DEFINE.WINDOW CURRENT &CUR 1 &CURS 80;+
0003700        DEFINE.WINDOW OLDCUR 21 1 2 61 PROTECT(YES);+
0003800        SCROLL.RIGHT OLDCUR AMOUNT(MAX);+
0003900        RESTORE.WINDOW CURRENT/
0004000                                                                 /* */
0004100 /* THE ENTRY WINDOW IS DELETED AND REDEFINED TO PRESERVE THE ORDER  */
0004200 /* OF THE WINDOWS ON THE SCREEN.                                   */
0004300 /* THE TEMPORARY AND PERMANENT CURSOR LOCATIONS ARE SET            */
0004400                                                                 /* */
0004500 /* DEFINE PF 1 TO RE-INVOKE THIS CLIST, ACCEPTING AS INPUT THE LINE */
0004600 /* (ROW) TO PLACE THE SPLIT AT. PRESSING PF 1 WITH NO INPUT WILL    */
0004700 /* RESTORE THE SCREEN AND PF KEYS TO THEIR ORIGINAL STATUS.         */
0004800                                                                 /* */
0004900 SMPUT /SAVE.WINDOW ENTRY;DELETE.WINDOW ENTRY;+
0005000        DEFINE.WINDOW ENTRY 23 1 1 120;RESTORE.WINDOW ENTRY;+
0005100        CHANGE.CURSOR 1 1 CURRENT TEMPORARY;+
0005200        CHANGE.CURSOR 1 1 ENTRY;+
0005300        CHANGE.PFK 1 'PUT ''RESTORE.SCREEN;RESTORE.PFK'' SMIN;+
0005400        PUT ''%SPLIT LINE(0¢1.)'' TSOIN' SMIN SUBSTITUTE(¢)/
```

Figure 2.5 - A TSO CLIST to Split the Screen

Assume the name of this CLIST is "SPLIT". To split the MAIN window at the tenth row, enter:

```
%split    [ENTER]
```

(This presumes implicit execution of the CLIST. See "OS/VS TSO Terminal User's Guide", GC28-0645, for more information.) Your screen should now look like Figure 2.6.

To move the split to the fifth line, enter:

5 [PF1]

The definition of PF 1 (which you just pressed) will restore the screen definition and the PF keys to their previous meanings (before the CLIST was first invoked) and then reexecute the CLIST. The CLIST will split the screen with the dashed line (the LINE window) placed at row 5. To return to the original screen layout, press:

[PF1]

This will restore the screen and PF keys, and the CLIST will be invoked with "LINE(0)" as the operand, which causes the CLIST to exit without splitting the screen.

Notice that the ENTRY window is saved, deleted, and redefined in the same place on the screen. While at first this may seem to be accomplishing nothing, it is really performing an important function. As was noted in "A Session Manager Primer",, on input, the display screen is read from top to bottom. This pertained to the IBM-supplied Default Screen layout. While each window on the screen is read from top to bottom, the windows are read in the order they were defined. If the ENTRY window had not been redefined in this CLIST, it would have been read before all the other windows, possibly causing some confusion if you entered data in more than one window at a time.

To determine the order the windows will be read on input, use the QUERY.TERMINAL or QUERY.WINDOW commands. The windows are listed in the order they are read.

Before you execute this CLIST for the first time, press PF 3 so that the Session Manager commands and error messages will be displayed in the CURRENT window.

If errors in the CLIST should disrupt the screen layout, press:

[CLEAR]

and then enter:

reset [ENTER]

to restore the IBM-supplied Default Screen layout.

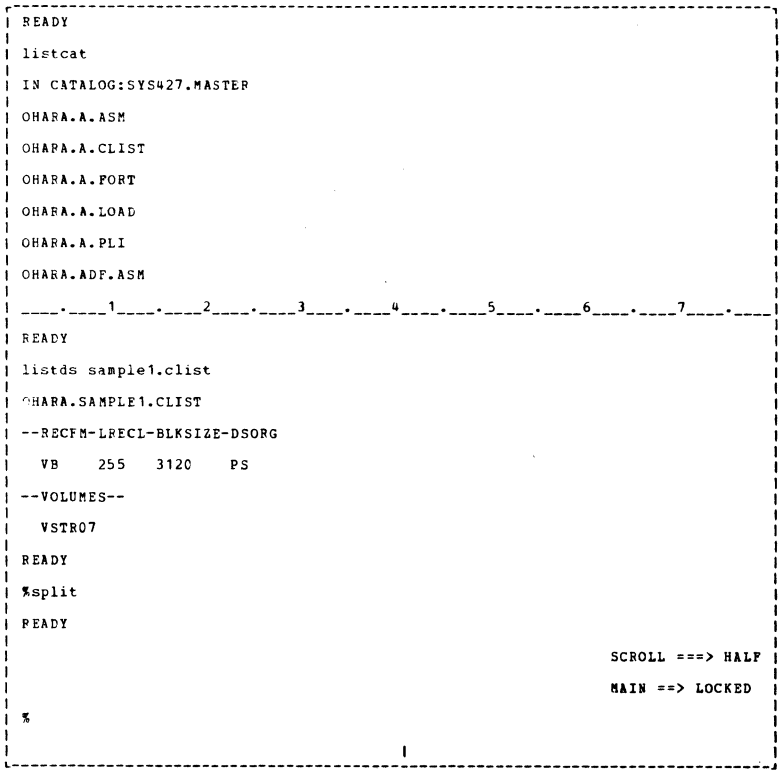


Figure 2.6 - After Execution of the "SPLIT" CLIST

Three Additional CLISTs

Three more CLISTS are presented here to give you an idea of how you can use the Session Manager commands to "customize" the display screen and PF key definitions to meet your particular needs.

The HSPLIT CLIST, shown in Figure 2.7, is similar to the SPLIT CLIST presented above. The comments in the CLIST outline how it redefines the display screen. The definition of PF 4 in this CLIST illustrates how the "extra" streams can be utilized to create status messages on the screen. Notice how the "LTITLE" window is scrolled to the right and left to display "MAIN" or "CURRENT" in the HEADER stream, thus indicating which window is the current Default Window. This allows you to easily control which window will be scrolled when you press one of the scrolling PF keys. The IBM-supplied Default Screen layout uses this same technique, described in "The Default Display Environment".

If you execute this CLIST, your screen should look similar to Figure 2.8.

```
0000100 PROC 0 LINE(10)
0000200 /*********************************************************************/
0000300 /* THIS CLIST SETS UP A "SPLIT - SCREEN" SESSION MANAGER SCREEN      */
0000400 /* LAYOUT. IT ASSUMES THE IBM - SUPPLIED DEFAULT SCREEN LAYOUT IS    */
0000500 /* THE CURRENT SCREEN LAYOUT WHEN IT IS INVOKED:                     */
0000600 /* - THE WINDOW "LINE" IS MOVED TO THE ROW SPECIFIED BY "LINE".      */
0000700 /* - THE WINDOW "CURRENT" IS EXPANDED TO FILL IN THE SPACE CREATED.  */
0000800 /* - THE WINDOW "SPACE" IS DEFINED TO FILL IN THE AREA ABOVE THE     */
0000900 /*       SCROLL AMOUNT VALUE IF NEEDED.                              */
0001000 /* - THE SCREEN AND PF KEYS ARE SAVED BEFORE THE SCREEN IS MODIFIED. */
0001100 /*********************************************************************/
0001200                                                                  /* */
0001300 /* EXIT IF THE LINE OPERAND IS OUT OF RANGE                          */
0001400                                                                  /* */
0001500 IF &LINE < 2 | &LINE > 21 THEN +
0001600   EXIT
0001700                                                                  /* */
0001800 /* COMPUTE THE SIZES OF THE WINDOWS TO BE DEFINED                    */
0001900                                                                  /* */
0002000 SET &TOPS = &LINE - 1
0002100 SET &BOT = &LINE + 1
0002200 SET &BOTS = 23 - &BOT
0002300 SET &BOTSX = 21 - &BOT
0002400                                                                  /* */
0002500 /* IS AN ADDITIONAL WINDOW (A FILLER WINDOW "SPACE") NEEDED?          */
0002600                                                                  /* */
0002700 IF &BOTSX > 0 THEN +
0002800   SET &WIN = DEF.W SPACE &BOT 63 &BOTSX 18 P(Y) V(HEADER);S.A 11 SPACE;
0002900 ELSE +
0003000   SET &WIN =
0003100                                                                  /* */
0003200 /* COMMENCE PROCESSING IN EARNEST                                    */
0003300                                                                  /* */
0003400 SMPUT /SAVE SCREEN;SAVE.PFK;+
0003500        SAVE.WIN MAIN;SAVE.WIN LINE;SAVE.WIN CURRENT;SAVE.WIN ENTRY;+
0003600        DEL.WIN MAIN;DEL.WIN LINE;DEL.WIN CURRENT;DEL.WIN ENTRY;+
0003700        DEFINE.WINDOW MAIN 1 1 &TOPS 80;+
0003800        DEFINE.WINDOW LINE &LINE 1 1 80;+
0003900        DEFINE.WINDOW CURRENT &BOT 1 &BOTS 62;+
0004000        DEFINE.WINDOW ENTRY 23 1 1 120;+
0004100        &WIN.+
0004200        RES.WIN ENTRY;RES.WIN CURRENT;RES.WIN LINE;RES.WIN MAIN;+
0004300        CHANGE.CURSOR &TOPS 1 MAIN TEMP;CHANGE.CURSOR 1 1 ENTRY;+
0004400        CHANGE.WINDOW MAIN OVERLAP(1) HOLD(0)/
0004500                                                                 /* */
0004600 /* DEFINE PF 1 TO RE-INVOKE THIS CLIST, ACCEPTING AS INPUT THE LINE  */
0004700 /* (ROW) TO PLACE THE SPLIT AT. PRESSING PF 1 WITH NO INPUT WILL     */
0004800 /* RESTORE THE SCREEN AND PF KEYS TO THEIR ORIGINAL STATUS.          */
0004900                                                                 /* */
0005000 /* PF 4 IS DEFINED TO SWITCH THE DEFAULT WINDOW BETWEEN THE "MAIN"   */
0005100 /* WINDOW AND THE "RIGHT" WINDOW: THUS SETTING WHICH WINDOW IS       */
0005200 /* SCROLLED WHEN PF 5, 7, 8, 10, 11, 12 IS PRESSED. ALSO, THE        */
0005300 /* "LTITLE" WINDOW IS MOVED TO DISPLAY THE NAME OF THE WINDOW THAT   */
0005400 /* IS CURRENTLY THE DEFAULT WINDOW. (THE NAMES WERE PREVIOUSLY       */
0005500 /* IN THE "HEADER" STREAM.                                           */
0005600                                                                 /* */
0005700 SMPUT /CHANGE.PF 1 'PUT ''RESTORE.SCREEN;RESTORE.PFKS'' SMIN;+
0005800               PUT ''%HSPLIT L(0¢1.)'' TSOIN' SMIN SUB(¢);+
0005900        CHANGE.PF 4 'CHANGE TERMINAL DEFAULT(CURRENT);SAVE.PF;+
0006000               SCROLL.LEFT LTITLE A(MAX);SCROLL.RIGHT LTITLE A(10);+
0006100               CHANGE.PF 4 ''CHANGE.TERMINAL DEFAULT(MAIN);+
0006200                     RESTORE.PF;SCROLL.LEFT LTITLE A(MAX)'' SMIN' +
0006300               SMIN;SCROLL.ABSOLUTE 10 LTITLE/
```

Figure 2.7 - The HSPLIT CLIST

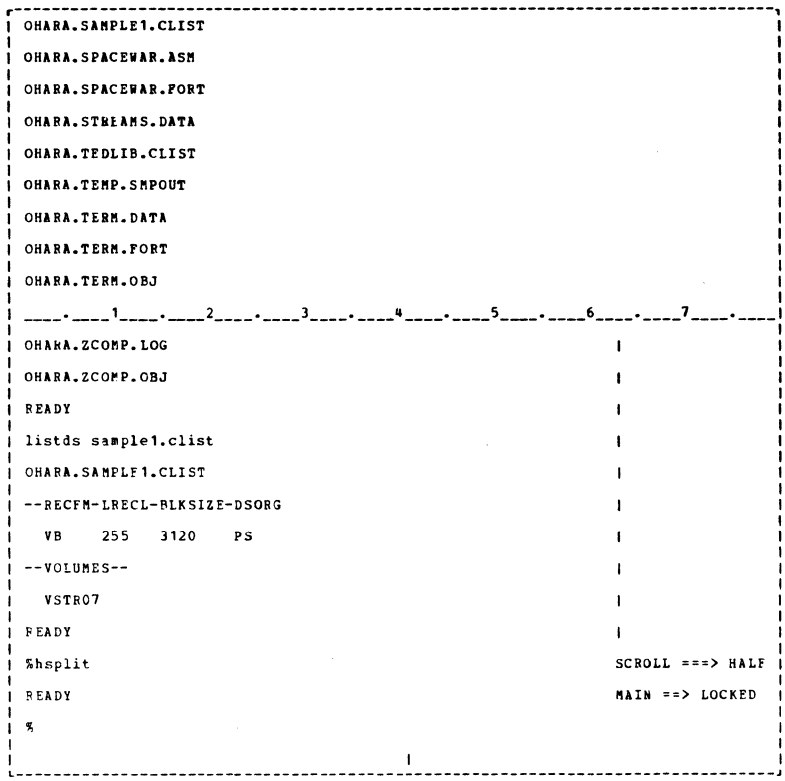


Figure 2.8 - The HSPLIT CLIST Has Been Executed

The VSPLIT CLIST, shown in Figure 2.9, is very similar to the HSPLIT CLIST. You may find this CLIST useful when testing programs or comparing members of partitioned data sets.

For example, you could use the LISTDS command to list the members of a partitioned data set, and then lock the MAIN window by pressing PF 10 (scrolling to the left when the window is already at the left border of the stream merely locks the window). Now execute the VSPLIT CLIST and then press PF 4 to make the RIGHT window become the Default Window. Pressing PF 12 will unlock the RIGHT window and you can now enter the LISTDS command for the second data set whose members you wish to display. By using PF 4 to change the Default Window, you can use the other PF keys to scroll the windows and compare the members of the data sets.

The screen layout created by executing this CLIST is shown in Figure 2.10.

```
0000100 PROC 0 COL(40)
0000200 /*********************************************************************/
0000300 /* THIS CLIST SETS UP A "SPLIT - SCREEN" SESSION MANAGER SCREEN      */
0000400 /* LAYOUT. IT ASSUMES THE IBM - SUPPLIED DEFAULT SCREEN LAYOUT IS    */
0000500 /* THE CURRENT SCREEN LAYOUT WHEN IT IS INVOKED:                     */
0000600 /* - THE "MAIN" WINDOW IS SPLIT VERITCALLY:                          */
0000700 /* - THE WINDOW "MAIN" IS REDEFINED WITH A NEW WIDTH OF "COL" - 1.   */
0000800 /* - A NEW WINDOW "SPLIT" IS CREATED TO PROVIDE A VISUAL INDICATION  */
0000900 /*   OF THE SPLIT. IT VIEWS THE "HEADER" STREAM IN WHICH VERTICAL    */
0001000 /*   BARS HAVE BEEN PLACED PREVIOUSLY.                               */
0001100 /* - THE WINDOW "RIGHT" IS DEFINED TO FILL IN THE AREA TO THE RIGHT  */
0001200 /*   OF THE "SPLIT" WINDOW. ITS ATTRIBUTES ARE FILLED IN BY          */
0001300 /*   RESTORING A COPY OF THE ATTRIBUTES OF THE "MAIN" WINDOW FROM    */
0001400 /*   THE "WINDOW" STACK.                                             */
0001500 /* - THE SCREEN AND PF KEYS ARE SAVED BEFORE THE SCREEN IS MODIFIED. */
0001600 /*********************************************************************/
0001700                                                                 /* */
0001800 /* EXIT IF THE COL OPERAND IS OUT OF RANGE                          */
0001900                                                                 /* */
0002000 IF &COL < 5 | &COL > 75 THEN +
0002100   EXIT
0002200                                                                 /* */
0002300 /* COMPUTE THE SIZES OF THE WINDOWS TO BE DEFINED                   */
0002400                                                                 /* */
0002500 SET &LEFTS = &COL - 1
0002600 SET &RIGHT = &COL + 2
0002700 SET &RIGHTS = 81 - &RIGHT
0002800                                                                 /* */
0002900 /* COMMENCE PROCESSING IN EARNEST                                   */
0003000                                                                 /* */
0003100 SMPUT /SAVE.SCREEN;SAVE.PFK;+
0003200        SAVE.WIN MAIN;SAVE.WIN MAIN;DELETE.WIN MAIN;+
0003300        DEFINE.WIN MAIN 1 1 19 &LEFTS;+
0003400        DEFINE.WIN SPLIT 1 &COL 19 2 HOLD(0) VIEW(HEADER) PROTECT(YES);+
0003500               SCROLL.ABSOLUTE 11 SPLIT;+
0003600        DEFINE.WIN RIGHT 1 &RIGHT 19 &RIGHTS;+
0003700        RESTORE.WIN MAIN;RESTORE.WIN RIGHT/
0003800                                                                 /* */
0003900 /* PF 4 IS DEFINED TO SWITCH THE DEFAULT WINDOW BETWEEN THE "MAIN"   */
0004000 /* WINDOW AND THE "RIGHT" WINDOW: THUS SETTING WHICH WINDOW IS       */
0004100 /* SCROLLED WHEN PF 5, 7, 8, 10, 11, 12 IS PRESSED. ALSO, THE       */
0004200 /* "LTITLE" WINDOW IS MOVED TO DISPLAY THE NAME OF THE WINDOW THAT   */
0004300 /* IS CURRENTLY THE DEFAULT WINDOW. (THE NAMES WERE PREVIOUSLY       */
0004400 /* IN THE "HEADER" STREAM.                                          */
0004500                                                                 /* */
0004600 /* DEFINE PF 1 TO RE-INVOKE THIS CLIST, ACCEPTING AS INPUT THE       */
0004700 /* COLUMN TO PLACE THE SPLIT AT. PRESSING PF 1 WITH NO INPUT WILL    */
0004800 /* RESTORE THE SCREEN AND PF KEYS TO THEIR ORIGINAL STATUS.         */
0004900                                                                 /* */
0005000 SMPUT /CHANGE.PF 4 'CHANGE.TERMINAL DEFAULT(RIGHT);SAVE.PF;+
0005100               SCROLL.LEFT LTITLE A(MAX);SCROLL.RIGHT LTITLE A(20);+
0005200               CHANGE.PF 4 ''CH.TERMINAL DEFAULT(MAIN);RESTORE.PF;+
0005300                      SCROLL.LEFT LTITLE A(MAX)'' SMIN' SMIN;+
0005400        SCROLL.ABSOLUTE 10 LTITLE;+
0005500        CHANGE.PF 1 'PUT ''RESTORE.SCREEN;RESTORE.PF'' SMIN;+
0005600               PUT ''%VSPLIT COL(0¢1.)'' TSOIN' SMIN SUB(¢)/
```

Figure 2.9 - The VSPLIT CLIST

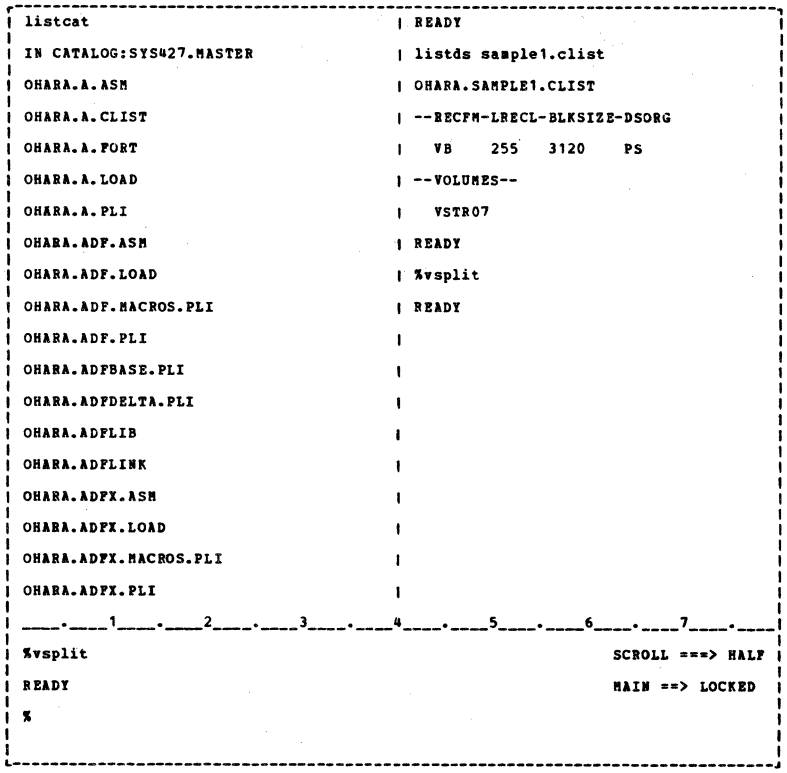


Figure 2.10 - The VSPLIT CLIST Has Been Executed

Both the HSPLIT CLIST and the VSPLIT CLIST depend on the SETUP CLIST, illustrated in Figure 2.11. This CLIST places data in the HEADER stream which is then viewed by windows defined in the two CLISTs. It also performs the useful function of allocating a partitioned data set of CLISTs called 'userid.CLIST' (where "userid" is your TSO userid) with a ddname of "SYSPROC". This allows you to implicitly execute any CLISTs in the data set, as described in "TSO Terminal User's Guide", GC28-0645.

Note: If you should issue the RESET command during your session, the data in the HEADER stream will be lost. You will then need to re-execute this CLIST to place the data there again.

```
0000100 /*********************************************************************/
0000200 /* THIS CLIST PLACES DATA IN THE "HEADER" STREAM FOR THE "SPLIT" AND */
0000300 /* "VSPLIT" CLISTS. IT SHOULD BE EXECUTED ONCE AT THE BEGINNING OF   */
0000400 /* THE SESSION.                                                      */
0000500 /* THIS CLIST ALSO ALLOCATES A PDS CLIST DATA SET WITH THE DDNAME OF */
0000600 /* "SYSPROC" FOR IMPLICIT EXECUTION OF OTHER CLISTS - AS DESCRIBED   */
0000700 /* IN "TSO TERMINAL USER'S GUIDE", GC28-0645.                        */
0000800 /*********************************************************************/
0000900                                                                  /* */
0001000 CONTROL NOMSG
0001100 SMPUT /PUT 'MAIN ==>  CURRENT>  RIGHT =>' HEADER I(2);+
0001200        PUT | HEADER I(2);PUT | HEADER I(2);PUT | HEADER I(2);+
0001300        PUT | HEADER I(2);PUT | HEADER I(2);PUT | HEADER I(2);+
0001400        PUT | HEADER I(2);PUT | HEADER I(2);PUT | HEADER I(2)/
0001500 SMPUT /PUT | HEADER I(2);PUT | HEADER I(2);PUT | HEADER I(2);+
0001600        PUT | HEADER I(2);PUT | HEADER I(2);PUT | HEADER I(2);+
0001700        PUT | HEADER I(2);PUT | HEADER I(2);PUT | HEADER I(2);+
0001800        PUT | HEADER I(2);PUT | HEADER I(2);PUT | HEADER I(2)/
0001900 FREE:
0002000 FREE FILE(SYSPROC)
0002100 ALLOCATE FILE(SYSPROC) DATASET('&SYSUID..CLIST') SHR
```

Figure 2.11 - The SETUP CLIST

HELP Information

HELP information is available for Session Manager commands by issuing:

```
HELP SM
```

The Session Manager TSO commands also have HELP information available. Simply issue the appropriate command variation:

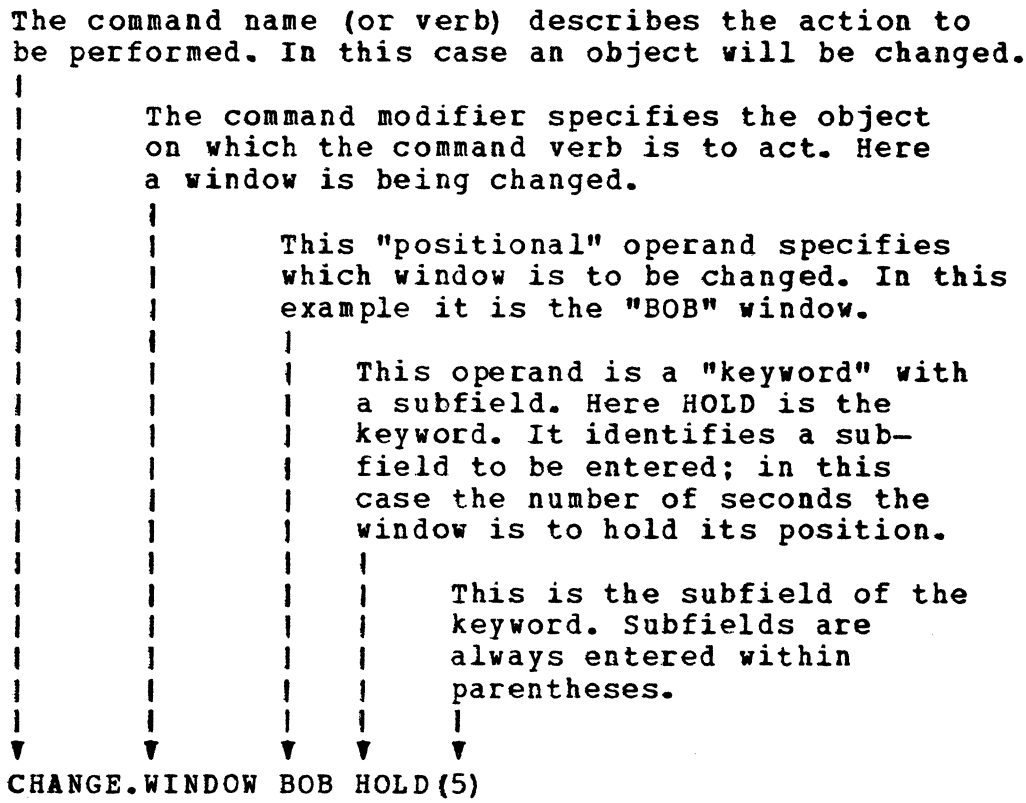
```
HELP SMCOPY  or  HELP SMC

HELP SMFIND  or  HELP SMF

HELP SMPUT   or   HELP SMP
```

SESSION MANAGER COMMAND SYNTAX

The Session Manager commands allow you to define, modify, and manipulate the operating characteristics of your display terminal. However, you don't have to learn them all to use the Session Manager.

Note: The syntax rules described here do not apply to the Session Manager TSO commands. See "TSO Commands".

A Session Manager command consists of a command name (a verb indicating the action performed by the command) followed, usually, by one or more operands. Operands supply the specific information needed for the command to execute. For example the CHANGE command has operands that specify what type of object to change, which object is to be changed, and what attributes of the object are to be changed:

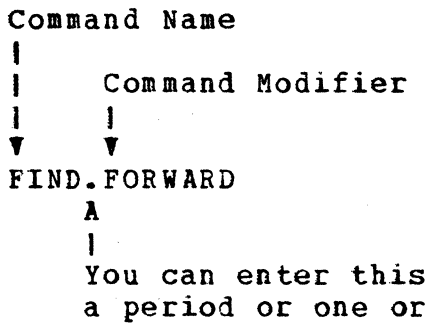


The above example illustrates the parts of a typical Session Manager command: the command name and modifier, the positional operand(s), and the keyword operand(s). You may enter each command name, modifier, or operand as it appears in this manual, or you may use an abbreviation, described in "Abbreviations".

Session Manager commands may be entered in upper or lower case, or a mixture of both and may not exceed 512 characters.

Command Names and Modifiers

A Session Manager command name is an alphabetic string up to 8 characters in length. Some of the command names are followed by "command modifiers". The FIND command is an example of this. A command modifier is also an alphabetic string up to 8 characters in length. It is separated from the command name by a period or one or more blanks:



The command modifier serves to qualify the action of the command name, much as data set names are qualified by additional names, separated by periods (although only a single command modifier may follow a command name). Each "command-name.modifier" sequence is treated as a separate command with its own operands.

Positional Operands

Positional operands follow the command name in a prescribed order. In this manual, the positional operands are shown in lower case letters. For example, the first positional operand of the CHANGE.PFK command is described as:

pfk-number

You must replace "pfk-number" with an integer number to specify the Program Function key to be changed when you enter the command.

In general, positional operands are the required operands of each command: you must enter them. If they are optional, they are shown enclosed in brackets in this manual, and they will have a default value, identified in the description of the operand. For example, the first positional operand of the SCROLL.FORWARD command is described as:

[pages]

You may enter the SCROLL.FORWARD command without specifying a "pages" operand. If you do, the default value of the operand (in this case "1") will be entered for you, just as if you had typed it. See "Defaults" for more information.

Keyword Operands

Keywords are names and symbols that have specific meaning to the Session Manager. You can include keywords in any order following the positional operands (if any). In this manual keywords are shown in uppercase. Here are two keyword operands of the CHANGE.TERMINAL command:

[ALARM(YES | NO)]

[CONTROL(seconds)]

They both are examples of keyword operands with subfield values. (The subfield values are enclosed in parentheses.) You must type the keyword or its abbreviation, a left parenthesis, and then the subfield value or values.

The description of the ALARM keyword shows the two possible values you may enter within the parentheses (in this case YES or NO). The values are shown separated by a vertical bar (the "or" symbol). This means the two values are mutually exclusive: enter either but not both.

The description of the CONTROL operand shows the subfield value in lower case. This means you are to substitute a value for the lower-case name. In this example you would enter a number to specifiy the number of seconds the keyboard may be locked.

The closing parenthesis need not be typed if it is the last character of the command.

Keyword operands are optional (indicated by the brackets in the syntax description). If there is a default value for the keyword or the subfield of the keyword, it is identified in the description of the keyword.

Defaults

In order to make the commands easier to enter, certain operands default to a specific value if you do not enter them with the command.

When a positional operand or a subfield of a keyword operand has a default value, the operand or subfield will be enclosed in brackets. The default value will be specified in the text description of the operand.

Many of the Session Manager commands operate on or refer to a specific window. To reduce the amount of typing needed, the concept of a "Default Window" is introduced. Whenever a window name is needed in a command, it may be omitted; the name of the "Default Window" will be substituted. (There is one exception to this rule: the DELETE command.)

You can specify which window of the screen layout is to be the Default Window using the CHANGE.TERMINAL command.

Abbreviations

You can enter command names, command modifiers, and keyword operands exactly as shown in this manual, or you can enter an abbreviation instead. Any command can be abbreviated by entering only those letters needed to distinguish it from other Session Manager commands. For example,the minimum abbreviation for the DELETE command is:

```
DEL
```

This distinguishes it from the DEFINE command.

In a similar manner, any operand can be abbreviated by entering only those letters needed to distinguish it from other operands of the command. For example, the keyword operands of the CHANGE.WINDOW command and their minimum abbreviations are:

ALARM	A
HOLD	H
OVERLAP	O
PROTECT	P
TARGET	T
UPDATE	U
VIEW	V

In this manual the minimum abbreviations for command names and modifiers are shown below the command name in the syntax description.

Because certain commands are more frequently used than others, they have been given special abbreviations. These commands and their abbreviations are:

DEFINE	D
SCROLL	S
RESTORE	R

These commonly used abbreviations are also accepted by the Session Manager:

CONTROL	CNTL
FORMAT	FMT

Acceptable Values for Positional Operands and Subfields

Session Manager window and stream names must be from 1 to 8 alphanumeric characters, with the first character alphabetic.

All numeric values must be integers between -99999 and 999999. Each command description specifies the numeric range acceptable for that command. If you enter an out-of-range numeric value, it will be rounded to the acceptable number nearest the entered value; no error message will be issued.

Text-strings (in the CHANGE.PFK, FIND and PUT commands) must be entered enclosed in single quotes. A single quote (') in the character string must be entered as two adjacent quotes ('').

If the text contains no imbedded blanks, commas, or parentheses, no enclosing quotes are needed, and single quotes in the text do not need to be repeated, as described above. If the text string is entered without enclosing quotes, however, it will be translated to upper-case. The text will always be converted to upper-case if execution is from a CLIST.

Entering Session Manager Commands

Session Manager commands may be entered from the keyboard, from the stored definition of a Program Function key (see the CHANGE PFK command), or from TSO CLISTs (see the SMPUT command). Regardless of the source, the following rules apply to the entry of Session Manager commands.

Multiple Session Manager commands may be entered on a line, separated by semicolons ";". The maximum number of characters on any one line (including blanks and semicolons) is 512. When multiple Session Manager commands are entered on a line, a syntax or exe-

cution error in one command will not prevent the remaining commands from executing.

Note: This can cause problems in TSO CLISTs when a command is dependent on execution of a previous Session Manager command.

All lower case letters are translated to upper case except as noted under "Acceptable Values", above.

Session Manager commands are executed from the SMIN stream. Thus the command you enter (from any of the above sources) must reach the SMIN stream for it to be executed. In the IBM-supplied default screen layout you must press PF 3 before you can enter Session Manager commands; after you have pressed it , you can then use PF 3 as an "enter" key for Session Manager commands. For more details, see "Using the Session Manager Commands".

Note: If the screen has been deleted (if all the windows have been deleted - see the DELETE.WINDOW command) any input entered from the keyboard will be lost. To recover from this situation, press the CLEAR key; you will be prompted to enter one or more Session Manager commands. The RESET command will restore the IBM-supplied default screen layout.

Syntax Conventions

The notation used to describe the Session Manager command syntax in this manual is outlined below:

- Each command is presented with one operand per line, with the operands separated by a blank line (whenever possible). If an operand does not fit on one line, it is continued on the next line.

- Any number of blanks or a single comma can be used to separate the parameters of a command.

- Lowercase letters or words represent variables for which you must substitute specific information as indicated in the command description.

- The symbols listed below should never be typed when entering a command; they are used only in this manual to describe the syntax.

brackets	[ ]
hyphen	-
underscore	_
vertical bar	|

- Brackets indicate that the operand within the brackets

is optional and can be omitted when you enter the command.

- Hyphens join lowercase words to form a single variable in the description. For example:

  pfk-number

  indicates you should enter the number of a Program Function key.

- Underscoring indicates that the keyword or value is the default for the operand.

- The vertical bar indicates alternatives. Only one of the operands can be selected. The possible choices for an operand are contained within the brackets or parentheses.

- Uppercase letters or words and the symbols listed below must be typed as shown in the statement description.

asterisk	*
parentheses	( )
period	.
single quote	'

- The asterisk is used in place of a name and means "all". For example:

  DELETE.WINDOW *

  means "delete all the windows".

- Parentheses are used to delimit the subfield of a keyword operand.

- The period is used as a delimiter between the command name and the command modifier.

  Note: The period can be replaced with one or more blanks: for example CHANGE.WINDOW can be entered as CHANGE WINDOW.

- The single quote is used to delimit text strings. A single quote (') in the character string must be entered as two adjacent quotes ('').

  If the text contains no imbedded blanks, commas, or parentheses, no enclosing quotes are needed, and single quotes in the text do not need to be repeated, as described above. If the text string is entered without enclosing quotes, however, it will be translated to upper-case.

ENVIRONMENT DEFINITION COMMANDS

The layout of the windows on the display screen, the attributes of those windows, the definitions of the Program Function keys, and some attributes relating to the terminal as a whole comprise your terminal "environment". The Session Manager Environment Definition commands allow you to dynamically define and modify your display terminal environment.

Note: Attributes set or changes made with these commands last only for the terminal session.

CHANGE.CURSOR

Use this command to change the location of the cursor on the display screen. There are two cursor "locations" you can establish with this command: a "permanent" location and a "temporary" location. You can specify whether the cursor location you give is to be a "permanent" or "temporary" location.

The cursor will return to the permanent location after each entry from the keyboard (after you press a PF, ENTER, Attention (PA1) or Cancel (PA2) key).

The cursor will move to the temporary location when this command is issued with the TEMPORARY keyword. It will stay there until after the next keyboard entry, at which time it will return to the permanent location.

In the IBM-supplied default screen layout the permanent cursor location is set to row 1, column 1 of the ENTRY window. A temporary cursor location is not set.

```
CHANGE.CURSOR   [row  column]
C      C
                [window-name]

                [TEMPORARY]
```

row column (positional)

Default: row - 1 column - 1

is the location (temporary or permanent) where the cursor is to return. It is specified relative to a window with "1 1" being the upper left-hand corner of the window. If you specify a row and column larger than the size of the window, or less than 1, the value will be adjusted to lie within the window. If the row value exceeds the size of the specified window, the value will be adjusted to the last row. If the column value is too large, it will be adjusted to the width minus 1.

window-name (positional)

Default: the Default Window

is the name of the window where the cursor is to be placed. If no window is specified, the cursor is placed in the Default Window. An exception to this is described under the TEMPORARY operand, below.

TEMPORARY (keyword)

Default: none

causes the cursor to move to its temporary location. If row and column or window-name are also entered, the command will set the temporary cursor location and move the cursor there. If both row and column and window-name are omitted, the command will simply move the cursor to the existing temporary location. If no temporary location has been set, the cursor is moved to the upper left-hand corner of the display screen.

Example:

Set the permanent location of the cursor to be row 5, column 3 of the BOB window:

```
change cursor 5 3 bob
```

The cursor will now return to this location after each entry from the keyboard.

Example:

Set the temporary location of the cursor to be row 7, column 1 of the JIM window, and move it there:

```
c cursor 7 1 jim temporary
```

The cursor will return to its permanent location after the next keyboard entry. To move the cursor to its temporary location, enter:

```
c c t
```

Example:

Change the permanent cursor location to row 1, column 1 of the Default Window:

```
change cursor
```

The cursor will now return to this location after each entry from the keyboard.

CHANGE.FUNCTION

Use this command to specify the input and output streams of the session "functions". These "functions" receive input from the keyboard and generate output which is placed in the streams. The functions are:

TSO — TSO has an input stream and an output stream. Data placed in the input stream is interpreted by TSO as commands or input to programs. The output produced by TSO commands and programs is placed in the output stream. You can also specify that the data placed in the TSO input stream is to be copied to the output stream when TSO reads it as input, thus creating in the output stream a complete record of your TSO session.

In the IBM-supplied default screen layout the input stream for TSO is TSOIN; the output stream is TSOOUT. Data placed in TSOIN is copied to TSOOUT and highlighted there (intensity of 2).

SM — The Session Manager also has an input stream and an output stream. Data placed in the input stream is interpreted by the Session Manager as Session Manager commands. The output produced by the Session Manager (mostly error messages) is placed in the output stream. You can also specify that the data placed in the the Session Manager input stream is to be copied to the output stream when the Session Manager reads it as input. This creates in the output stream a complete record of Session Manager commands you entered and any error messages they produced.

In the IBM-supplied default screen layout the input stream for the Session Manager is SMIN; the output stream is SMOUT. Data placed in SMIN is copied to SMOUT but not displayed (intensity of 0).

MSG — You can receive messages from other TSO users, from the operator, and status messages about background jobs you submit. These messages are placed in an output stream. There is no input stream for the Message function, and thus no copy stream.

In the IBM-supplied default screen layout the output stream for these messages is the TSOOUT stream; the messages are placed there highlighted (intensity of 2), and the alarm is sounded when a message is received. Thus the messages are interleaved with your TSO session input and output.

There are three types of Session Manager streams which can be specified as input sources for, or output targets from, the three

functions:

INPUT These streams serve as input buffers to TSO or the Session Manager. The two input streams are TSOIN and SMIN.

OUTPUT These streams serve as the output buffers for TSO, the Session Manager, and messages from other TSO users or the Operator. The output streams are MESSAGE, SMOUT, EXTRA1, and TSOOUT.

EXTRA These streams are, as the name implies, "extra" streams; they serve neither as input or output to any function. They are useful for copying data to (via the SMCOPY command) or for containing the output of certain Session Manager commands (QUERY and SNAPSHOT). Two of these extra streams (the HEADER and EXTRA3 streams) are used in the default screen layout to hold the dashed numbered line seen on the screen and to show the scroll amount. The extra streams are EXTRA2, EXTRA3, and HEADER.

You can use the QUERY.STREAMS command to display the names and attributes of all the streams.

```
CHANGE.FUNCTION   function
C      F
                  [ALARM(INPUT | OUTPUT | NO)]

                  [COPY(copy-stream-name [intensity]) |
                   NOCOPY ]

                  [INPUT(input-stream-name)]

                  [OUTPUT(output-stream-name [intensity])]
```

function (positional)

Default: none

is the name of the session function to be changed. Valid functions are:

MSG is used to specify the stream where messages from other TSO users, the operator and background jobs will be placed. Only an output stream can be specified for MSG; all other keywords will be ignored.

SM is used to specify the streams associated with the Session Manager. An input, output and copy stream can be specified.

TSO is used to specify the streams associated with TSO. An input, output and copy stream can be specified.

ALARM(INPUT | OUTPUT | NO) (keyword)

Default: none

specifies whether or not the display terminal's audible alarm is to sound when the function gets data from its input stream or places data in its output stream. This is independent of whether or not a window is viewing the stream.

Note: If your terminal does not have an audible alarm, this command will still be accepted, but the alarm will never sound, of course. (The Session Manager has no way of knowing whether or not your terminal has an alarm.)

INPUT specifies that the alarm will sound when the function gets a line of data from its input

stream.

OUTPUT specifies that the alarm will sound when the function places data in its output stream.

NO specifies that actions by the function will not cause the alarm to sound.

COPY(copy-stream-name intensity) (keyword)

Default: copy-stream-name - none
intensity - 1

names the stream to which data placed in an input stream of one of the functions (specified as INPUT, below) will be copied to. The stream must be of the OUTPUT type.

For example, at the start of your session, TSOIN is the input stream for the TSO function, TSOOUT the output stream. Data placed in TSOIN will be copied to TSOOUT and highlighted there. The copy occurs when the function reads the line of data from the input stream. This causes the copied input to be interleaved with the output in the output stream, creating a complete record of your session.

"Intensity" specifies the brightness at which the copied data is to be displayed. Valid values are:

0 The copied data is not to be displayed. (You can see the line occupied by the copied data but the data itself is invisible.) See the examples for a use of copying with 0 intensity.

1 The copied data is to display at normal intensity.

2 The copied data is to be highlighted.

This operand is mutually exclusive with the NOCOPY operand.

NOCOPY (keyword)

Default: none

specifies that data placed in the input stream will not be copied to another stream. This operand is mutually exclusive with the COPY operand.

INPUT(input-stream-name) (keyword)

Default: none

names the stream that will serve as an input buffer for

the specified function. The stream must be of the INPUT type.

OUTPUT(output-stream-name intensity) (keyword)

Default: output-stream-name - none
intensity - 1

names the stream that will serve as an output buffer for the specified function. The stream must be of the OUTPUT type.

"Intensity" specifies the brightness at which the output data is to be displayed. Valid values are:

1 The data is to display at normal intensity.

2 The data is to be highlighted.

Example:

Cause the Session Manager to get commands from the SMIN stream, and place any output (error messages) in the SMOUT stream, highlighted:

```
change function sm input(smin) output(smout 2) nocopy
```

The SMOUT stream will contain only Session Manager error messages, displayed at high intensity.

Now have the Session Manager commmands in SMIN copied to SMOUT:

```
change function sm copy(smout)
```

Copies of the Session Manager commands will now be interleaved in the SMOUT stream with the error messages.

Example:

Cause Session Manager error messages to be placed in the TSOOUT stream:

```
c f sm o(tsoout)
```

This is useful to do when you are debugging TSO CLISTs containing Session Manager commands; you can see the lines of the CLIST and any error messages they caused all in the TSOOUT stream. Examples of TSO CLISTs containing Session Manager commands are given in "Using the Session Manager Commands".

Example:

Have Session Manager error messages highlighted in the SMOUT

stream, and Session Manager commands copied there but not displayed (0 intensity):

```
c f sm o(smout 2) copy(smout 0)
```

If you have a one-line window viewing the SMOUT stream, the above command will cause the window to always be blank (displaying non-display data) until a Session Manager error message is issued. When the next Session Manager command is issued, the error message will be replaced by a non-display line (the copied command). This is done in the IBM-supplied default screen layout.

Example:

Have TSO get its input from TSOIN and place its output in TSOOUT, highlighted. Also have data in TSOIN copied to TSOOUT at normal intensity (this is just the reverse of the default):

```
c f tso i(tsoin) o(tsoout 2) c(tsoout 1)
```

Example:

Set the alarm to sound when you receive a message from another TSO user, and have the message placed in the TSOOUT stream, highlighted:

```
change function msg output(tsoout 2) alarm(output)
```

This is done in the IBM-supplied default screen layout.

CHANGE.PFK

Use this command to change the definition of a Program Function (PF) key). The Program Function keys were originally defined for you in the IBM-supplied default screen layout. PF keys are defined as text strings that will be placed in the specified target stream when the PF key is pressed.

You can, in effect, make a PF key a TSO command (or commands) by defining the text string as a valid TSO command or subcommand and specifying TSOIN as the target stream. This is useful under TSO EDIT or TEST: PF keys can be defined as often-entered subcommands such as SAVE or GO.

You can, in effect, make a PF key a Session Manager command by defining the text string as a valid Session Manager command and specifying SMIN as the target stream. This is done in the IBM-supplied default screen layout to allow you to use the PF keys for scrolling.

If you have the character string sent to another stream such as the TSOOUT stream, the PF key can be used to comment your session journal in that stream.

When the PF key is pressed, the following occurs:

1. Each line of data you typed on the screen since you last pressed the ENTER, CLEAR, PA1, PA2, or any PF key is placed in the next available line in the target stream of the window from which the line came on the display screen. (Just as if you had first pressed the ENTER key.) See the CHANGE.WINDOW command to set the target stream for a window.

   Note: The order in which the Session Manager reads the lines you have typed on the screen is discussed in "Using the Session Manager Commands".

2. The text-string of the PF key definition is placed in the target-stream of the PF key definition.

Optionally, you can specify symbolic arguments in the definition text string. These arguments will be replaced by data from the screen when the PF key is pressed. The SUBSTITUTE operand controls this function.

For example, you could specify 'change /&1./&2./' as the definition text string for a PF key, to be sent to the TSOIN stream. If you typed:

```
abc xyz
```

and pressed the PF key, this EDIT subcommand would be executed:

```
CHANGE /abc/xyz/
```

```
CHANGE.PFK      pfk-number
C      P
                definition-text-string

                target-stream-name

                [SUBSTITUTE[(identifier  [delimiter])]]
```

pfk-number (positional)

Default: none

is the number of the Program Function key to be defined. On the IBM 3270 Display Terminal this number must be from 1 to 24.

Note: If the display terminal has only 12 Program Function keys (or none), this command will be accepted, but you won't be able to use the PF key definitions for which you have no keys, because you won't be able to press them. (The Session Manager has no way of knowing how many Program Function keys your terminal has.)

definition-text-string (positional)

Default: none

is the character string to be placed in the target-stream. The string normally must be enclosed in single quotes. A single quote (') in the character string must be entered as two adjacent quotes (''). If the text contains no imbedded blanks, commas, or parentheses, no enclosing quotes are needed, and single quotes in the text do not need to be repeated. If the definition-text-string is entered without enclosing quotes or if the command is executed out of a CLIST, however, it will be translated to upper-case.

The definition-text-string can be a Session Manager or TSO Command, input to an application program, or any other string of characters.

The text string can contain symbolic arguments for which the Session Manager substitutes data entered immediately before the PF key is pressed. This is described under the SUBSTITUTE operand, below.

target-stream-name (positional)

Default: none

is the name of the Session Manager stream where the text-string is to be placed.

SUBSTITUTE(identifier delimiter) (keyword)

Default: identifier - &
delimiter - ¢

specifies that the data read from the screen by the Session Manager when the PF key is pressed is to be substituted into the definition-text-string of the PF key, replacing symbolic arguments. The text-string thus constructed is then placed in the target-stream of the PF key.

The text string can contain symbolic arguments for which the Session Manager substitutes data entered immediately before the PF key is pressed. These arguments are of the form

&1., &2., &3., ...&n., and &*.

The "&" character above represents the "identifier" subfield of the SUBSTITUTE operand. Any character can be used as the symbolic argument identifier. If the identifier character appears elsewhere in the definition-text-string, it must be doubled (as for quotes, above).

The periods immediately following the numeric arguments are used to distinguish the arguments from the characters immediately following them in the definition-text-string. The periods are not needed if the character after the argument is not a digit.

The symbolic arguments above represent the 1st, 2nd, 3rd, and nth tokens of each data line on the display screen. The "*" argument represents the remainder of the field after tokenization of that line has stopped (see below).

Symbolic substitution for <u>each</u> line of data found on the screen when the PF key was pressed is performed as follows:

- The line of data is scanned once from left to right and broken into as many "tokens" (words) as there are numbered symbolic arguments in the PF key definition text-string. The tokens are separated by the "<u>delimiter</u>" subfield of this operand. Normally, the delimiter is a blank. One or more blanks are treated as a single delimiter. This process is called "tok-

enization".

The remainder of the text in the line becomes the "*" token. If there are no numeric symbolic arguments in the PF key definition text-string, the entire line of data becomes the "*" token.

- The tokens are substituted into the PF key definition text-string, replacing the symbolic arguments. If there are fewer tokens than arguments, null characters are substituted for those arguments lacking tokens. The symbolic arguments are prefixed in the PF key definition text string by the "identifier".

- The resulting text string is placed in the next available line of the target stream (at the bottom of the stream).

The above processing is repeated for each line of data you typed on the screen before pressing the PF key.

Note: If there are no modified fields on the screen, null characters are substituted for each argument in the text-string, and the string is placed in the target stream. No commands may be entered via a PF key with symbolic substition because tokenization will occur.

Example:

When PF 1 is pressed, have it issue the TSO TIME command.

```
change pfk 1 'time' tsoin
```

Example:

Set PF 12 to issue the QUERY.TERMINAL command, directing the output to the EXTRA1 stream, and then cause the Default Window to view that stream:

```
c pfk 12 'query terminal extra1;change w view(extra1)' smin
```

Example:

When PF 2 is pressed, have it issue the TSO LISTDS command, and treat each line typed just before the key is pressed as the data set name operand of the command:

```
c pfk 2 'listds &1..* members' tsoin substitute
```

If you type the following lines on the screen:

```
margaret
bob
```

when you press PF 2 the following TSO commands will be executed:

```
LISTDS margaret.* MEMBERS
LISTDS bob.* MEMBERS
```

Note: The first period following "&1" in the definition text-string indicated that the substituted token was to be concatenated with the remaining text.

Since the "*" was not preceded by a "&", it was not interpreted as a symbolic argument.

Example:

Define PF 4 so when a number is typed and PF 4 is pressed, the Default Window will be scrolled forward that number of pages:

```
c p 4 'scroll forward +1' smin sub(+)
```

Typing "5" and pressing PF 4 will generate the following Session Manager command:

```
SCROLL FORWARD 5
```

Pressing PF 4 without typing anything will generate:

```
SCROLL FORWARD
```

Note: The plus sign (+) was used as the symbolic argument identifier. This is useful when the command is to be placed in a TSO CLIST; it avoids confusion with the ampersands (&) of the CLIST.

Example:

Set PF 9 so if you enter a line on the screen that begins with a question mark (?) and press PF 9, the line will be treated as the fully qualified data set name operand of the DELETE command:

```
change pfk 9 'delete ''&2''' tsoin s(& ?)
```

You can use the TSO LISTCAT command to display the data set names of your catalog. To delete one or more data sets, insert a "?" in front of those data sets you wanted deleted, and press PF 9. A TSO DELETE command will be generated for each data set name you typed a question mark in front of.

Note: Multiple quotes were used in the text-string and the question mark was used as a token delimiter.

Example:

Define PF 7 to place "one & two & ..." (where "..." is whatever you type when you press PF 7) in the TSOOUT stream:

```
ch pf 7 'one && two && &*.' tsoout sub
```

If you type:

```
three & four
```

and press PF 2, the following line will be placed in the TSOOUT stream:

```
one & two & three & four
```

Note: Doubled ampersands (&) were used as the ampersand was the identifier character (by default).

For additional examples, see "Using the Session Manager Commands" and "The Default Display Environment".

CHANGE.STREAM

Use this command to modify attributes of Session Manager streams. The attributes set remain in effect for the terminal session unless modified by subsequent use of this command. The streams and their attributes were originally set for you in the default environment.

You can use the QUERY.STREAMS command to display the names and attributes of all the streams.

```
CHANGE.STREAM      stream-name
C      S
                   [ALARM(YES | NO)]

                   [CLEAR]
```

stream-name (positional)

Default: none

is the name of the Session Manager stream to be changed. Use the QUERY.STREAM command to list the streams and their attributes.

ALARM(YES | NO) (keyword)

Default: none

specifies whether or not the display terminal's audible alarm is to sound when data is placed in the stream. This is independent of whether or not a window is viewing the stream.

Note: If your terminal does not have an audible alarm, this command will still be accepted, but the alarm will never sound, of course. (The Session Manager has no way of knowing whether or not your terminal has an alarm.)

CLEAR (keyword)

Default: none

erases all the data in the stream. The stream itself is not deleted.

Example:

Erase all the data in the TSOOUT stream:

```
change stream tsoout clear
```

Example:

Cause the terminal alarm to sound whenever data is placed in the EXTRA1 stream.

```
change stream extra1 alarm(yes)
```

In the IBM-supplied default screen layout, PF 4, when pressed, places a snapshot of the display screen in the EXTRA1 stream. Thus the alarm would now sound whenever you pressed PF 4.

CHANGE.TERMINAL

Use this command to define or modify attributes of your display environment that apply to the terminal as a whole. The attributes set remain in effect for the terminal session unless modified by subsequent use of this command or the RESTORE command (see "Session Control Commands"). The "Default Window", set by the DEFAULT operand, is used as a default value in other Session Manager commands when a window operand is not entered. Examples of its use in a "Split Screen" are given in "Using the Session Manager Commands".

In order for the Session Manager to update your display screen, TSO requires that the terminal keyboard remain locked. Unfortunately, this prevents you from typing while the Session Manager is updating the screen. To let you control when the keyboard is to be locked (and the screen updated), the Session Manager provides a timer, set by the CONTROL operand.

If you wish to wait for each TSO command to execute and have its output displayed before entering the next command (the way 3270 display terminals operate without the Session Manager), set CONTROL to a very high value (up to 999).

If you wish to enter either TSO or Session Manager commands without first waiting for the currently executing command to complete, set CONTROL to 0. The keyboard will unlock immediately after the Session Manager rewrites the display screen, at which time you can enter new commands without further delay.

If the keyboard unlocks before the output of a command has been displayed, you will not see the output until you press the ENTER key or one of the PF keys. (This tells the Session Manager to update the screen if there is new data to display.)

By setting CONTROL to a higher value (say, between 10 and 20) it is possible to strike a workable compromise between an unlocked keyboard and an up-to-date screen. This will allow most TSO commands time to execute and have their output displayed, yet not lock up the terminal keyboard for long periods of time, preventing you from entering additional input. For longer running commands (compilations, for example) set CONTROL to an even higher number (say 30 to 60) to allow the screen to be updated as soon as there is new output to display.

The use of the CONTROL operand is further described in "Using the Session Manager Commands".

```
CHANGE.TERMINAL     [ALARM(YES | NO)]
C      T
                    [CONTROL(LAST | seconds)]

                    [DEFAULT(window-name)]
```

ALARM(YES | NO) (keyword)

Default: none

specifies whether or not the display terminal's audible alarm is to sound when the terminal keyboard unlocks and is ready for input.

Note: If your terminal does not have an audible alarm, this command will still be accepted, but the alarm will never sound, of course. (The Session Manager has no way of knowing whether or not your terminal has an alarm.)

CONTROL(LAST | seconds) (keyword)

Default: none
Alternate Form: CNTL

sets a timer to unlock the terminal keyboard after the specified number of seconds has elapsed. "Seconds" must be an integer from 0 to 999.

Entering LAST sets the control timer to the last non-zero value you entered for CONTROL (in a previous entry of this command).

Note: The IBM-supplied Default Screen layout sets CONTROL to 15 initially, so if the first time you enter this command with CONTROL(LAST), it is the same as entering CONTROL(15).

The Session Manager can only update the display screen while the keyboard is locked. This operand lets you specify how long the Session Manager should keep the keyboard locked while updating the display screen or "waiting" for more output (from TSO).

If more data enters a stream being viewed by an unlocked window, the window will move forward to display the new data (according to the settings of the HOLD, UPDATE, and OVERLAP operands of the CHANGE.WINDOW command for that window).

Note: This operand specifies the maximum time the keyboard is to remain locked. If TSO is not currently processing a command and if all the unlocked windows on the screen are at the bottom of the stream they are viewing, the keyboard will unlock immediately.

DEFAULT(window-name) (keyword)

Default: none

names a window as the "Default Window." The Default Window is used by the other Session Manager commands as the default value for "window-name" operands when a window name is not entered with the command. In a similar fashion, the SCROLLing PF keys (7-12) reference the Default window.

Example:

Set the terminal so the keyboard will never be locked for more than 10 seconds:

```
change.terminal control(10)
```

Now allow the keyboard to be locked for up to 60 seconds (while a program is compiling, perhaps):

```
c t c(60)
```

Example:

Define the terminal so each time the keyboard unlocks the alarm will sound:

```
c t a(y)
```

CHANGE.WINDOW

Use this command to modify the attributes of an existing window on the display screen. Use the QUERY.WINDOWS or QUERY.TERMINAL command to display the names and attributes of the windows on the display screen.

```
CHANGE.WINDOW     [window-name]
C      W
                  [ALARM(YES | NO)]
                  [HOLD(INPUT | seconds)]
                  [OVERLAP(lines)]
                  [PROTECT(YES | NO)]
                  [TARGET(stream-name  [intensity])]
                  [UPDATE(LINE | NEWEST | PAGE)]
                  [VIEW(stream-name)]
```

window-name **(positional)**

Default: the Default Window

is the name of the window whose attributes are to be changed.

ALARM(YES | NO) **(keyword)**

Default: none

specifies whether or not the display terminal's audible alarm is to sound when the Session Manager moves the window to display new data in the stream it is viewing.

The alarm will not sound when you move the window via the FIND or SCROLL command.

Note: If your terminal does not have an audible alarm, this command will still be accepted, but the alarm will never sound, of course. (The Session Manager has no way of knowing whether or not your terminal has an alarm.)

HOLD(INPUT | seconds) **(keyword)**

Default: none

specifies how often the window (when unlocked) is to be moved toward the newest data in the stream it is viewing.

Note: this operand has effect only when the window is unlocked: windows locked by scrolling commands remain in place until unlocked or moved by additional scrolling commands. See "Screen Control Commands".

INPUT specifies that the window (when unlocked) be moved toward the newest data in the stream each time you supply input by pressing the ENTER or any PF key. The normal action of the key (executing a command, etc.) is performed.

seconds specifies that the window (when unlocked) be moved toward the newest data in the stream; the Session Manager will hold the window in place the specified number of seconds between each move. During the time the window pauses the keyboard will remain locked. The keyboard will be unlocked (and movement of the window will cease) when the window has reached the newest data in the stream.

"Seconds" must be an integer from 0 to 999.

The settings of the CONTROL operand of the CHANGE.TERMINAL command will interrupt this process and unlock the keyboard when the specified number of seconds has expired.

OVERLAP(lines) (keyword)

Default: none

specifies how many lines of the window's old position should be repeated in the new position. Thus as the window moves over the stream toward the newest data, some lines will be displayed twice.

"Lines" must be an integer from 0 to 999. If "lines" is greater than or equal to the number of lines in the window, it is replaced by a value 1 less than the number of lines in the window.

PROTECT(YES | NO) (keyword)

Default: none

specifies whether or not the window is to serve as a

data entry area on the screen. A protected window cannot be used to enter data; the keyboard will lock if you try to type in it. (Use the RESET key to unlock the keyboard.) An unprotected window can serve as an entry area for commands or other input. Use the TARGET operand, described below, to specify where the input will be sent.

TARGET(stream-name intensity) (keyword)

Default: stream-name - none
intensity - 1

specifies which stream will receive input data entered in this window, and at what intensity that data in the stream will be displayed.

When you enter or modify data displayed in a window and press ENTER or a PF key not involved with symbolic substitution, the lines you modified are read by the Session Manager. "Stream-name" specifies which stream is to receive these changed lines.

"Intensity" lets you specify the brightness at which the data in the target-stream is to be displayed. Valid values are:

0 The data is not to be displayed. This is used in the IBM-supplied default screen layout to create a window in which passwords can be entered.

1 The data is to display at normal intensity.

2 The data is to be highlighted.

UPDATE(LINE | NEWEST | PAGE) (keyword)

Default: none

specifies how much new data must enter the stream to trigger a rewrite (update) of the window.

While the window is unlocked, it moves over the stream it is viewing toward the newest data in the stream. (How fast it moves is specified by the HOLD operand, above.) When the window reaches the newest data in the stream (the bottom of the stream), it stops and waits for new data to enter the stream. Once the new data has entered the stream, the window is no longer at the bottom of the stream and will again move toward the newest data.

UPDATE specifies how much new data must enter the stream before the window again moves over the stream, and

whether the window will display every new line or skip to the newest data as it moves over the stream. This second aspect of UPDATE is illustrated in Figure 3.1.

LINE specifies that the window, when unlocked, moves sequentially over the stream it is viewing toward the newest data in the stream. Thus all of the new data will be displayed as the window moves over the stream (in contrast to NEWEST, explained below). New lines entering the stream will be displayed in the window until the window is full. At this time, the window will move forward (repeating the number of lines specified by the OVERLAP operand) and the new data will again start to fill up the window.

NEWEST specifies that the window, when unlocked, will always display the newest data in the stream it is viewing. When new data enters the stream the window is moved directly to the bottom of the stream. Some data in the stream may be skipped over. Thus if a large amount of data is sent to the stream, only the last few lines (the number of lines in the window) will be displayed. The Session Manager scrolling commands can be used to view the data that was skipped over.

PAGE specifies that the window, when unlocked, moves sequentially over the stream it is viewing toward the newest data in the stream. Thus all of the new data will be displayed as the window moves over the stream (in contrast to NEWEST, explained above). The window is only moved, however, when there are enough new lines of data minus the number of overlap lines to fill the window. If there are not enough lines in the new data to fill the window, the window will not move to display the data until enough additional data enters the stream to fill the window (a "page" of data).

VIEW(stream-name) (keyword)

Default: none

specifies which stream will be displayed in the window. The window will initially be placed over the newest data in the stream (at the bottom of the stream), and will be unlocked.

Example:

Make window JIM view the SMOUT stream:

```
change.window jim view(smout)
```

Example:

Make window LARRY serve as an entry area for TSO passwords (the password will not be displayed as you type it):

```
c.w larry target(tsoin 0)
```

Example:

Make the Default Window unavailable for input (protected from data entry):

```
c w p(y)
```

Note: Here the name of the window to be changed was not entered, thus the Default Window was changed.

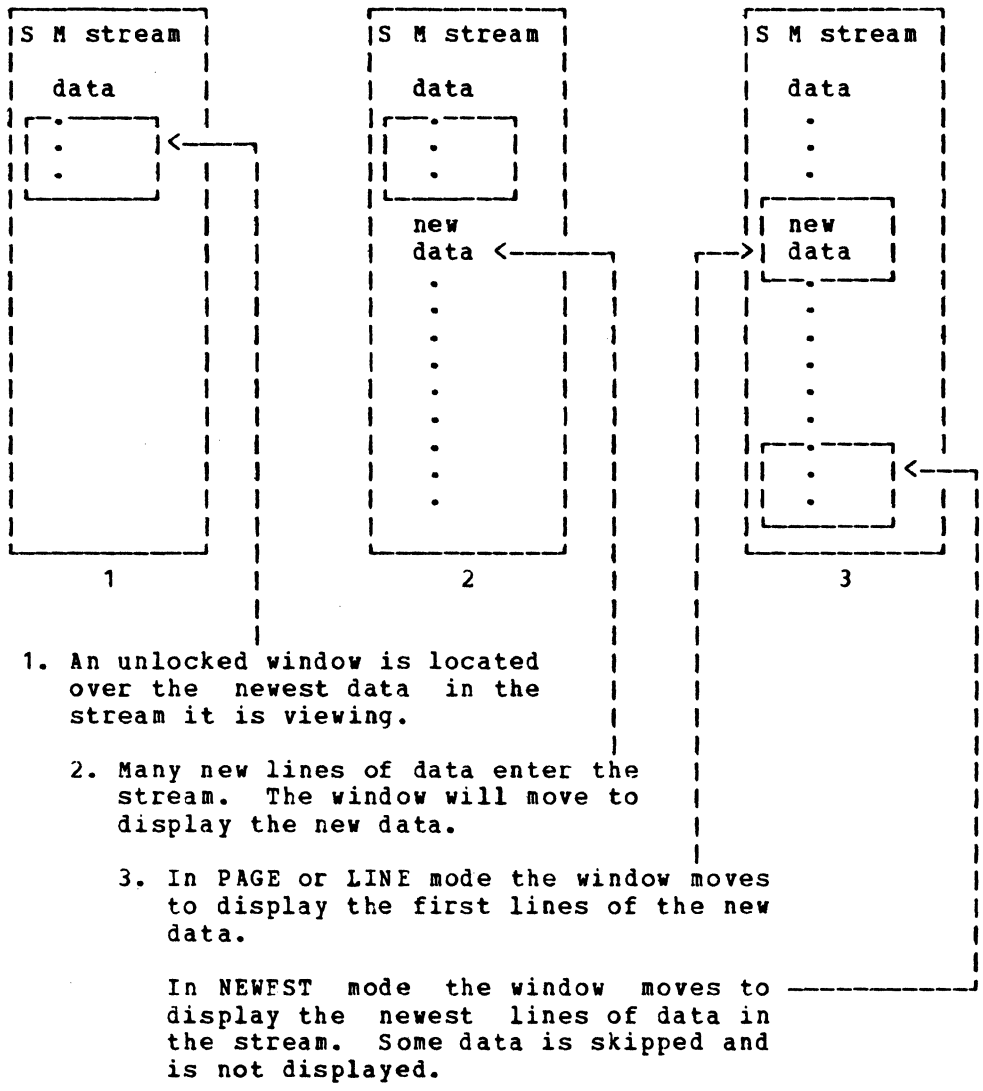


Figure 3.1 - Action of the UPDATE Operand

DEFINE.WINDOW

Use this command to define a new window on the display screen.

A newly defined window cannot exceed the physical limits of the display screen, with one exception, noted below. Also, a new window to be defined on the display screen cannot overlap the physical location of an existing window. Use the DELETE.WINDOW command to remove existing windows, if necessary.

When you define a new window its attributes remain in effect until you modify them with the CHANGE.WINDOW command. When defining a new window, you can give it the desired attributes all at once with the operands of this command, or you can issue this command followed by one or more CHANGE.WINDOW commands to build up the attributes of the window. The default attributes for a new window are:

- ALARM(NO)
- HOLD(0)
- OVERLAP(1)
- PROTECT(NO)
- TARGET(TSOIN 1)
- UPDATE(LINE)
- VIEW(TSOOUT)

Note: A window that is 1 line deep and starts in column 1 may exceed the width of the display screen (that is, it may have a width greater than 80 columns).

The "ENTRY" window of the IBM-supplied default screen layout is an example of this. It has a width of 120 characters. While visually the window is continued on the next line of the display, it is still a single line window. The window may continue to "wrap" the screen width for as many rows as there are room for; however it may not wrap from the last row to the 1st row.

Note: The possible screen sizes for the IBM 3270 Display Terminals are as follows:

Models 2 and 12 - 24 rows X 80 columns = 1920
Models 3 and 13 - 32 rows X 80 columns = 2560
Models 4 and 14 - 43 rows X 80 columns = 3440

```
DEFINE.WINDOW     window-name
D      W
                  row
                  column
                  lines
                  width
                  [ALARM(YES | NO)]
                  [HOLD(INPUT | seconds)]
                  [OVERLAP(lines)]
                  [PROTECT(YES | NO)]
                  [TARGET(stream-name [intensity])]
                  [UPDATE(LINE | NEWEST | PAGE)]
                  [VIEW(stream-name)]
```

window-name (positional)

Default: none

is the name of the window being defined. It must be from 1 to 8 alphanumeric characters, the first character must be alphabetic.

row (positional)

Default: none

specifies which row of the display screen the top line of the window is to occupy. The "row" must be an integer between 1 and either 24, 32, or 43 depending upon the screen size of the terminal.

column (positional)

Default: none

specifies which column of the display screen the left edge of the window is to occupy. The "column" must be an integer between 1 and 80.

lines (positional)

Default: none

specifies the number of lines in the window. (The height of the window.) The "lines" must be an integer between 1 and either 24, 32, or 43 depending upon the screen size of the terminal.

width (positional)

Default: none

specifies the width of the window in character positions. The "width" must be an integer between 1 and 80, unless "column" and "lines" are 1, in which case it can be an integer up to 1920, 2560, or 3440 depending upon the screen size of the terminal. See the note at the beginning of the description of this command.

ALARM(YES | NO) (keyword)

Default: no

specifies whether or not the display terminal's audible alarm is to sound when the Session Manager moves the window to display new data in the stream it is viewing. The alarm will not sound when you move the window via the FIND or SCROLL command.

Note: If your terminal does not have an audible alarm, this command will still be accepted, but the alarm will never sound, of course. (The Session Manager has no way of knowing whether or not your terminal has an alarm.)

HOLD(INPUT | seconds) (keyword)

Default: 0

specifies how often the window (when unlocked) is to be moved toward the newest data in the stream it is viewing.

Note: this operand has effect only when the window is unlocked: windows locked by scrolling commands remain in place until unlocked or moved by additional scrolling commands. See "Screen Control Commands".

INPUT specifies that the window (when unlocked) be moved toward the newest data in the stream each time you supply input by pressing the ENTER or any PF key. The normal action of the key (executing a command, etc.) is performed.

seconds specifies that the window (when unlocked) be

moved toward the newest data in the stream; the Session Manager will hold the window in place the specified number of seconds between each move. During the time the window pauses the keyboard will remain locked. The keyboard will be unlocked (and movement of the window will cease) when the window has reached the newest data in the stream.

"Seconds" must be an integer from 0 to 999.

The setting of the CONTROL operand of the CHANGE.TERMINAL command will interrupt this process and unlock the keyboard when the specified number of seconds has expired.

OVERLAP(lines) (keyword)

Default: 1

specifies how many lines of the window's old position should be repeated in the new position. Thus as the window moves over the stream toward the newest data, some lines will be displayed twice.

"Lines" must be an integer from 0 to 999. If "lines" is greater than or equal to the number of lines in the window, it is replaced by a value 1 less than the number of lines in the window.

PROTECT(YES | NO) (keyword)

Default: no

specifies whether or not the window is to serve as a data entry area on the screen. A protected window cannot be used to enter data; the keyboard will lock if you try to type in it. (Use the RESET key to unlock the keyboard.) An unprotected window can serve as an entry area for commands or other input. Use the TARGET operand, described below, to specify where the input will be sent.

TARGET(stream-name intensity) (keyword)

Default: TSOIN
intensity - 1

specifies which stream will receive input data entered in this window, and at what intensity that data in the stream will be displayed.

When you enter or modify data displayed in a window and press ENTER or a PF key, the lines you modified are read

by the Session Manager. "Stream-name" specifies which stream is to receive these changed lines.

"Intensity" lets you specify the brightness at which the data in the target-stream is to be displayed. Valid values are:

0 The data is not to be displayed. This is used in the IBM-supplied default screen layout to create a window in which passwords can be entered.

1 The data is to display at normal intensity.

2 The data is to be highlighted.

UPDATE(LINE | NEWEST | PAGE) (keyword)

Default: line

specifies how much new data must enter the stream to trigger a rewrite (update) of the window.

While the window is unlocked, it moves over the stream it is viewing toward the newest data in the stream. (How fast it moves is specified by the HOLD operand, above.) When the window reaches the newest data in the stream (the bottom of the stream), it stops and waits for new data to enter the stream. Once the new data has entered the stream, the window is no longer at the bottom of the stream and will again move toward the newest data.

UPDATE specifies how much new data must enter the stream before the window again moves over the stream, and whether the window will display every new line or skip to the newest data as it moves over the stream. This second aspect of UPDATE is illustrated in Figure 3.1.

LINE specifies that the window, when unlocked, moves sequentially over the stream it is viewing toward the newest data in the stream. Thus all of the new data will be displayed as the window moves over the stream (in contrast to NEWEST, explained below). New lines entering the stream will be displayed in the window until the window is full. At this time, the window will move forward (repeating the number of lines specified by the OVERLAP operand) and the new data will again start to fill up the window.

NEWEST specifies that the window, when unlocked, will always display the newest data in the stream it is viewing. When new data enters the stream

the window is moved directly to the bottom of the stream. Some data in the stream may be skipped over. Thus if a large amount of data is sent to the stream, only the last few lines (the number of lines in the window) will be displayed. The Session Manager scrolling commands can be used to view the data that was skipped over.

PAGE specifies that the window, when unlocked, moves sequentially over the stream it is viewing toward the newest data in the stream. Thus all of the new data will be displayed as the window moves over the stream (in contrast to NEWEST, explained above). The window is only moved, however, when there are enough new lines of data minus the number of overlap lines to fill the window. If there are not enough lines in the new data to fill the window, the window will not move to display the data until enough additional data enters the stream to fill the window (a "page" of data).

VIEW(stream-name) (keyword)

Default: TSOOUT

specifies which stream will be displayed in the window. The window will initially be placed over the newest data in the stream (at the bottom of the stream), and will be unlocked.

Example:

Create a screen layout having an output window of occupying the top 22 lines of the display screen, and an input window that occupies the bottom two lines of the display screen but is logically a single line:

```
define window output 1 1 22 80
define window input 23 1 1 160 view(header)
change.terminal default(output)
change.cursor 1 1 input
scroll.absolute 3 input
```

Note: The last three commands complete the screen layout by setting the OUTPUT window as the Default Window (so that scrolling commands to move it may be more easily issued), positioning the cursor, and moving the INPUT window to view a "%" in the HEADER stream.

For additional examples of this command, see "Using the Session Manager Commands" and "The Default Display Environment".

DELETE.WINDOW

Use this command to delete a window from the screen. The space occupied by the window is then available to be defined as another window or windows.

Note: when all the windows on the display screen have been deleted, command entry from the keyboard is not possible unless the CLEAR key is pressed. See "A Session Manager Primer" and "Entering Session Manager Commands".

```
DELETE.WINDOW     window-name | *
DEL    W
```

window-name (positional)

Default: none

names the window to be deleted.

* (positional)

Default: none

specifies that all the windows on the screen are to be deleted.

Example:

Delete a window on the screen named JIM:

```
delete.window jim
```

Example:

Delete all the windows on the display screen:

```
del.w *
```

SCREEN CONTROL COMMANDS

The Screen Control commands are the Session Manager commands that let you move a window over the stream it is viewing. Note that to "move" a window is not to change its physical position on your display screen, but instead to change which data it is viewing in the stream. (To rearrange the windows on your display screen, see the DEFINE.WINDOW command.)

Locked and Unlocked Windows

When a window is first defined as viewing a stream (via the DEFINE WINDOW command), it is unlocked: if no scrolling commands are ever issued, the arrival of new data in the stream causes the Session Manager to move the window toward the newest data in the stream, until it reaches the newest data (the bottom of the stream. Once a scrolling command (FIND or SCROLL) is issued, the window is locked, which means the Session Manager will not automatically move the window over the stream. The window will now move only when scrolling commands are issued by the user. See "A Session Manager Primer" for more information.

The UNLOCK command returns a window to the unlocked state, and the Session Manager will again automatically move the window over the stream.

FIND

Use this command to search for a specific character string in the stream the specified window is viewing. Once the character string is found, the Session Manager will scroll (move) the window so that the found text string is the top line displayed in the window.

After the window is moved, it is locked in position and will remain there until moved via another scrolling command or the UNLOCK command.

If the text string is not found, the window is not moved (and not locked) and a message is placed in the Session Manager output stream.

```
FIND.BACKWARD     text-string
F    B
                  [window-name]

FIND.FORWARD      text-string
F    F
                  [window-name]

FIND.LINE         [window-name]
F    L
                  TARGET(stream-name)
```

BACKWARD (modifier)

starts the search at the first line in the stream above the line viewed as the top line of the window and proceeds backward to the top of the stream.

FORWARD (modifier)

starts the search at the first line in the stream below the line viewed as the top line of the window and proceeds forward to the bottom of the stream.

LINE

finds the line number of the line being viewed in the top of the specified window and writes a message with the line number to the specified TARGET. This allows you an easy method for finding a line number for a subsequent SCROLL.ABSOLUTE command or for SMCOPYing a range of lines. (The window is not locked by FIND.LINE.)

text-string (positional)

Default: none

is the set of characters to be searched for. If the text-string contains blank characters, commas, or parentheses, it must be enclosed in single quotes. A quote in the text-string must be represented as two single quotes.

If there are no blanks, commas, or parentheses in the text-string the enclosing quotes may be omitted. In this case, however, the text-string will be translated to uppercase before the search commences. When operating from a CLIST, translation to uppercase occurs even if the text is quoted.

window-name (keyword)

Default: the Default Window

is the name of the window that is to be scrolled to the text-string. The stream that this window is viewing is the stream that will be searched for the text-string. For FIND.LINE, this is the name of the window whose top line number is found and placed in a message written to the target stream.

TARGET (stream-name)

Default: SM output stream name

The following message is written to the specified stream:

ADF031I window-name VIEWING LINE nnnnnn

Example:

The default window is viewing the TSOOUT stream. It is located at the bottom of the stream. Find the last time (most recent occurence) you entered the LINK command:

```
find.backward link
```

Now that the window has scrolled to where you entered the LINK command, find the next time you edited a data set called abc.asm:

```
find.forward 'edit abc.asm'
```

Now find the end of the edit session (the next "READY" message):

```
f f READY
```

SCROLL

This command will cause the Session Manager to move a window over the stream it is viewing. After the window has been moved it is locked in that position, and will remain fixed until moved via another scroll command or by the UNLOCK command.

SCROLL.ABSOLUTE S A	line-number [window-name]
SCROLL.BACKWARD S B	[pages] [lines] [window-name] [AMOUNT(HALF | MAX | PAGE | amount)]
SCROLL.FORWARD S F	[pages] [lines] [window-name] [AMOUNT(HALF | MAX | PAGE | amount)]
SCROLL.LEFT S L	[columns] [window-name] [AMOUNT(HALF | MAX | PAGE | amount)]
SCROLL.NEWEST S N	[window-name]
SCROLL.OLDEST S O	[window-name]
SCROLL.RIGHT S R	[columns] [window-name] [AMOUNT(HALF | MAX | PAGE | amount)]

ABSOLUTE (modifier)

specifies that the Session Manager is to move the specified window so that the line number specified is the top line displayed in the window. Use the QUERY command or the SMFIND TSO command to find specific line numbers.

BACKWARD (modifier)

specifies that the Session Manager is to move the specified window backward (up) over the stream toward the oldest data (top of the stream).

Note: The data in the window will appear to move down. The limit for scrolling backward is the top of the stream.

FORWARD (modifier)

specifies that the Session Manager is to move the window forward (down) over the stream toward the newest data (bottom of the stream).

Note: The data in the window will appear to move up. The limit for scrolling forward is the bottom of the stream.

LEFT (modifier)

specifies that the Session Manager is to move the specified window to the left over the stream.

Note: The data in the window will appear to move to the right. The limit for scrolling to the left is the left border of the stream (column 1).

NEWEST (modifier)

specifies that the Session Manager is to move the specified window to the newest data in the stream (to the bottom of the stream).

OLDEST (modifier)

specifies that the Session Manager is to move the specified window to the oldest data in the stream (to the top of the stream).

RIGHT (modifier)

specifies that the Session Manager is to move the specified window to the right the specified number of character columns (positions).

Note: The data in the window will appear to move to the left. The limit for scrolling to the right is the 32,768th column position.

line-number (positional)

Default: current location of window

specifies the line number of the stream the window is to be scrolled to.

If you enter a value of 0 or less, line-number will be set to 1 (the oldest data). If you enter a value greater than the highest line number in the stream, line-number will be set to the highest line number in the stream (the newest data).

window-name (positional)

Default: the Default Window

is the name of the window to be moved.

pages (positional)

Default: 1 if AMOUNT keyword not entered
0 if AMOUNT keyword entered

specifies how many pages to move the window. A page is defined as the number of lines in the window.

If you enter a negative value, the window will scroll in a direction opposite to the one specified by the command modifier. Values that would cause the window to scroll beyond the oldest or newest data in the stream will be adjusted to place the window at the oldest or newest data (depending on the direction of the scrolling).

lines (positional)

Default: 0

specifies how many lines to move the window.

If you enter a negative value, the window will scroll in a direction opposite to the one specified by the command modifier. Values that would cause the window to scroll beyond the oldest or newest data in the stream will be adjusted to place the window at the oldest or newest data (depending on the direction of the scrolling).

columns (positional)

Default: 40 if AMOUNT keyword not entered

0 if AMOUNT keyword entered

specifies the number of columns to move the window.

If you enter a negative value, the window will scroll in the opposite direction specified by the command modifier. Values that would cause the window to scroll beyond the left edge of the stream will be adjusted to place the window at the left edge of the stream. Values that would cause the window to scroll beyond column 32768 will be adjusted to place the window at column 32768.

AMOUNT(HALF | MAX | PAGE | amount) (keyword)

Default: none

specifies the amount the window is to be scrolled. It may be specified instead of or in addition to the positional operands "columns", "lines", and "pages". If you enter a value for one of the above positional operands and a value for AMOUNT, the two values are summed, and the window is scrolled the resulting amount. Valid AMOUNT values are:

HALF specifies that the window is to be scrolled half a page. For forward or backward scrolling, a page is defined as the number of lines in the window. For scrolling to the right or left, a page is defined as the number of columns in the window.

MAX specifies that the window is to be scrolled to the maximum possible value. For forward scrolling this is the bottom of the screen (equivalent to "SCROLL.NEWEST"). For backward scrolling this is the top of the stream (equivalent to "SCROLL.OLDEST"). For scrolling to the right, the window will be scrolled to column 32768. For scrolling to the left, the window will be scrolled to column 1.

PAGE specifies that the window is to be scrolled a full page. For forward or backward scrolling, a page is defined as the number of lines in the window. For scrolling to the right or left, a page is defined as the number of columns in the window.

amount specifies how many lines or columns to move the window. The range of permissible values is the same as for the "lines" operand when scrolling forward or backward; as for the "columns" operand when scrolling to the left or to the right.

Example:

Scroll the Default Window to the oldest data:

```
scroll.oldest
     or
scroll.backward amount(max)
```

Example:

Scroll the POK window forward 1 page:

```
s.f pok
  or
s.f pok a(p)
```

Example:

Scroll the Default Window right 40 columns:

```
s r
```

or (assuming the window is 80 columns wide):

```
s r a(half)
```

Example:

Scroll the Default Window to the leftmost edge of the stream it is viewing:

```
scroll left 99999
     or
scroll left amount(m)
```

Example:

Scroll the CATHY window backward 20 lines:

```
s b 0 20 cathy
     or
s b cathy a(20)
```

Example:

Scroll the LARRY window forward 2 pages less 5 lines:

```
s f 2 -5 larry
     or
s f 2 larry a(-5)
```

UNLOCK

This command unlocks the specified window. When a Session Manager scrolling command is executed, the window that was moved is "locked", that is, the window will no longer be automatically moved by the Session Manager toward the newest data in the stream. Only user-initiated Session Manager scrolling commands will move a locked window.

After this command is issued, the window is "unlocked" and the Session Manager will again automatically move the window forward over the stream, toward the newest data in the stream.

```
UNLOCK.HERE       [window-name]
U      H

UNLOCK.NEWEST     [window-name]
U      N

UNLOCK.RESUME     [window-name]
U      R
```

HERE (modifier)

unlocks the window in its current location.

NEWEST (modifier)

moves the window to display the newest data in the stream; then unlocks it.

RESUME (modifier)

moves the window to display what it was displaying when it was last unlocked; then unlocks it.

window-name (positional)

Default: the Default Window

specifies the name of the window to be unlocked.

Example:

Move the Default Window to the bottom of the stream it is viewing (to the newest data) and unlock it:

```
unlock newest
```

Example:

Scroll back the Default Window three pages, then move it to where it was when you started scrolling back and unlock it:

```
s b 3

un.resume
```

Example:

Unlock window NORM in its current location:

```
u h norm
```

SESSION CONTROL COMMANDS

The remaining Session Manager commands are called the Session Control commands.

END

Use this command to end Session Manager display support of your TSO session. Your TSO session will continue with standard display support. If you wish to have Session Manager display support of your terminal after having issued this command, it is necessary to issue the TSO LOGON command.

Note: The data in your streams will be lost when this command is issued. Use the SMCOPY TSO command to save data in the streams.

```
END
```

PUT

Use this command to place a string of characters in a Session Manager stream. If the target-stream specified is the TSOIN stream, the characters will be interpreted as a TSO command, and will be executed by TSO. If the target-stream is the SMIN stream, the characters will be interpreted as a Session Manager command and the Session Manager will execute it.

The text-string is placed in the next available line in the stream (at the bottom of the stream).

```
PUT                 text-string
P
                    target-stream-name

                    [INTENSITY(intensity)]
```

text-string (positional)

Default: none

is the character string to be sent to the target-stream, enclosed in single quotes. A single quote (') in the text-string must be entered as two adjacent quotes (''). A null text string of two adjacent quotes ('') may be specified.

If the text-string contains no imbedded blanks, commas or parentheses, no enclosing quotes are needed, and single quotes in the text do not need to be repeated, as described above. If the definition-text-string is entered without enclosing quotes or if the command is executing out of a CLIST, however, it will be translated to upper-case.

Note: Any Session Manager command cannot be greater than 512 characters in length.

target-stream-name (positional)

Default: none is the name of the Session Manager stream where the text-string is to be placed. The text will be placed at the bottom of the stream.

INTENSITY(intensity) (keyword)

Default: 1

is the brightness at which the text-string, when placed in the target-stream, is to be displayed. Valid values are:

0 The data is not to be displayed.

1 The data is to be displayed at normal intensity.

2 The data is to be displayed at high intensity (highlighted).

Example:

Place a comment in the TSOOUT stream, highlighted:

```
put 'This is a comment' tsoout int(2)
```

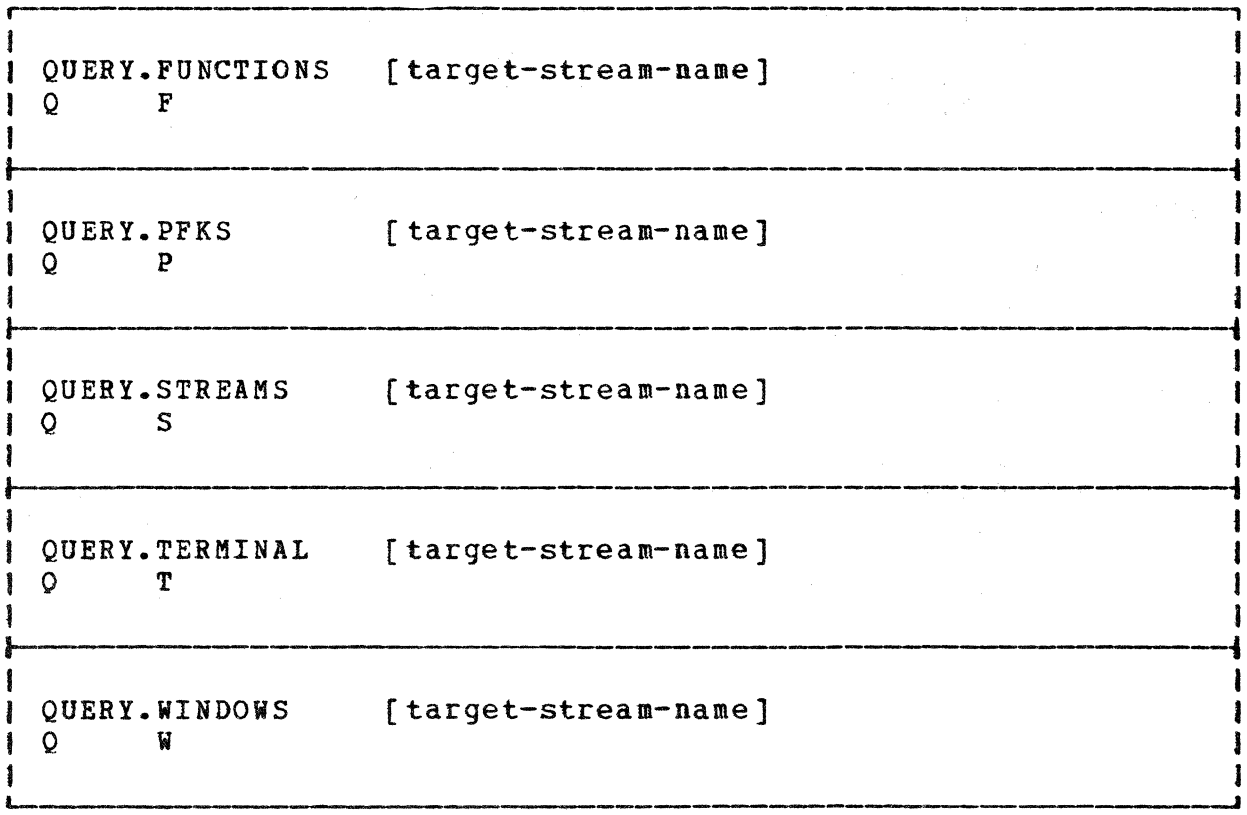
Example:

Use the PUT command in the definition-text-string of the CHANGE.PFK command to define a PF key to issue both a TSO command and a Session Manager command:

```
c.p 3 'put time tsoin;put ''unlock newest'' smin' smin
```

QUERY

This command displays information about portions of the current Session Manager environment.



FUNCTIONS (modifier)

displays the session functions currently defined and their input, output, and copy streams, and alarm settings.

PFKS (modifier)

displays the current definition of all the Program Function keys.

STREAMS (modifier)

displays the status of each stream: how large it is, how many lines have been used, whether the data in it has wrapped, and other attributes.

TERMINAL (modifier)

displays the name of each window on the screen and its attributes as set in the CHANGE.WINDOW command. It also

displays the status of the terminal as set in the CHANGE.TERMINAL command.

WINDOWS (modifier)

displays the position of each window on the display screen and its logical position on the stream it is viewing.

target-stream-name (positional)

Default: TSOOUT

is the stream where the output of this command is to be placed. The output is in the form of a table as shown below, so it is advisable to direct the output to a stream with a sufficiently large window to view it.

Example:

Sample output from the QUERY.FUNCTION command is shown in Figure 3.2. There is one entry for each of the session functions. The output in Figure 3.2 shows that the input stream for TSO is TSOIN, and that the alarm will not be sounded when TSO gets input from the stream. When a line is given to TSO by the Session Manager, it is also copied to the TSOOUT stream, and displayed there at high intensity ("INT" of 2). Output from TSO is placed in the TSOOUT stream and displayed at normal intensity ("INT" of 1).

The MSG function has no input or copy stream, so "*NONE*" is displayed in those columns.

```
FUNCTION      INPUT              OUTPUT              COPY
  NAME      STREAM  ALARM   STREAM  INT ALARM   STREAM  INT
TSO         TSOIN       N   TSOOUT    1     N   TSOOUT    2
SM          SMIN        N   SMOUT     2     N   SMOUT     1
MSG          *NONE*         TSOOUT    2     N    *NONE*
QUERY COMPLETE
```

Figure 3.2 - Sample Output From the QUERY.FUNCTIONS Command

Sample output from the QUERY.PFKS command is shown in Figure 3.3. These PF key definitions were defined for use with the TSO EDIT command. There is one entry for each of the Program Function keys:

- The number of the PF key is displayed.

- The stream in which the text string will be placed is displayed.

- The identifier character and delimiter character are displayed if the PF key accepts input as substitutable arguments; that is, only if the SUBSTITUTE operand was specified when the CHANGE.PFK command was entered for that key.

- The text of the PF key is displayed as it was entered in the CHANGE.PFK command (quotes are shown doubled). You may need to scroll to the right to view all of the text string.

```
PFK NO.  STREAM   ID DELIM   TEXT
PFK  1   TSOIN              'TOP'
PFK  2   TSOIN              'END NOSAVE'
PFK  3   SMIN               'CHANGE.CURSOR 10 9 TEMPORARY'
PFK  4   SMIN               'PUT ''UP 18'' TSOIN;PUT ''LIST * 19'' TSOIN' SMIN
PFK  5   SMIN      ¢        'PUT ''UP ¢1.'' TSOIN;PUT ''LIST * 19'' TSOIN' SMIN
PFK  6   TSOIN     ¢        'FIND /¢*./'
PFK  7   SMIN               'PUT ''DOWN 18'' TSOIN;PUT ''LIST * 19'' TSOIN' SMIN
PFK  8   SMIN      ¢        'PUT ''DOWN ¢1.'' TSOIN;PUT ''LIST * 19'' TSOIN' SMIN
PFK  9   TSOIN     ¢        'SAVE ¢*.'
PFK 10   TSOIN              'BOTTOM'
PFK 11   TSOIN     ¢        'CHANGE /¢1./¢2./'
PFK 12   SMIN               'UNLOCK.NEWEST'
QUERY COMPLETE
```

Figure 3.3 - Sample Output From the QUERY.PFKS Command

Sample output from the QUERY.STREAMS command is shown in Figure 3.4. There is one entry for each of the streams:

- The name of the stream is displayed.

- The line range the stream presently encompasses. The "LOW" line is the line number of the oldest data in the stream. If "LOW" is 1, the stream has not yet wrapped. The "HIGH" line is the line number of the newest data in the stream. For example, the SMIN stream shown in Figure 3.4 has wrapped.

- The maximum size of the stream in lines and bytes is shown.

- The number of lines and bytes currently used by the stream is shown. When either of these numbers exceeds the maximum the stream will wrap.

- The type of stream is shown and the alarm setting of the stream is shown.

Note: Streams and "wrapping" are discussed in "A Session Manager Primer", and stream types are described in the CHANGE.FUNCTION command.

```
 STREAM    LINE RANGE    MAXIMUM SIZE          USED
  NAME     LOW    HIGH   LINES     BYTES     LINES     BYTES    TYPE     ALARM
TSOIN        1      37     155      4096        37      2659    INPUT     NO
TSOOUT       1     269    4005    147456       269     15509    OUTPUT    NO
EXTRA1       1       1     405     32768         1        38    OUTPUT    NO
SMOUT        1      27     155      4096        27      1777    OUTPUT    NO
HEADER       1       5      55      1024         5       292    EXTRA     NO
EXTRA3       1       1     105      1024         1        38    EXTRA     NO
EXTRA2       1       1     105      1024         1        38    EXTRA     NO
MESSAGE      1       1     105      1024         1        39    OUTPUT    NO
SMIN        29      58     155      4096        30      4020    INPUT     NO
QUERY COMPLETE
```

Figure 3.4 - Sample Output From the QUERY.STREAMS Command

Sample output from the QUERY.TERMINAL command is shown in Figure 3.5.

The first two lines of the output displays values set in the CHANGE.TERMINAL command: the CONTROL timer and the ALARM setting of the terminal keyboard.

The next two lines display the current number of windows defined on the display screen, and the maximum number of windows permitted. The name of the Default Window is displayed (set in the CHANGE.TERMINAL command) as well as the permanent cursor position (set in the CHANGE.CURSOR command).

The rest of the output contains one entry for each of the windows currently defined on the screen. The attributes of each window, as set in the DEFINE.WINDOW and CHANGE.WINDOW commands, are displayed.

```
KEYBOARD     CNTL    ALARM
               0       N
WINDOWS      CURRENT #   MAXIMUM #       DEFAULT WINDOW     CURSOR POSITION
               10          25                MAIN              ENTRY     1   1
  NAME       VIEW    LOCKED  PROT  TARGET  INTENSITY  ALARM  HOLD  OVERLAP  UPDATE
MAIN       TSOOUT      Y       N    TSOIN       1        N      1        9      L
LINE       HEADER      Y       Y    TSOIN       1        N      0        0      N
CURRENT    TSOOUT      N       N    TSOIN       1        N      0        0      N
STITLE     HEADER      Y       Y    TSOIN       1        N      0        1      L
SVALUE     EXTRA3      N       Y    TSOIN       1        N      0        1      L
LTITLE     HEADER      Y       Y    TSOIN       1        N      0        1      L
LVALUE     HEADER      Y       Y    TSOIN       1        N      0        1      L
ENTRY      HEADER      Y       N    TSOIN       1        N      0        0      N
VLINE      HEADER      Y       Y    TSOIN       1        N      0        0      N
PASSWD     SMOUT       N       N    TSOIN       0        N      0        0      N
QUERY COMPLETE
```

Figure 3.5 - Sample Output From the QUERY.TERMINAL Command

Sample output from the QUERY.WINDOW command is shown in Figure 3.6. There is one entry for each of the windows currently defined on the display screen:

- The name of the window, and its screen location is displayed (set in the DEFINE.WINDOW command).
- The stream the window is viewing is displayed (set in the DEFINE.WINDOW and CHANGE.WINDOW commands).
- The next three entries display the position of the window on the stream it is viewing:
  - "PRESENT" displays the lines of the stream the window is currently viewing.
  - "RESUME" displays the lines of the stream the window will display if the UNLOCK.RESUME command is issued.
  - "NEWEST" displays the lines of the stream the window will display if the UNLOCK.NEWEST command is issued.

						PRESENT		RESUME		NEWEST	
					VIEWING	TOP	BOTTOM	TOP	BOTTOM	TOP	BOTTOM
WINDOW	ROW	COL	LINES	WIDTH	STREAM	LINE	LINE	LINE	LINE	LINE	LINE
MAIN	1	1	19	80	TSOOUT	52	70	70	88	70	88
LINE	20	1	1	80	HEADER	2	2	10	10	32	32
CURRENT	21	1	2	62	TSOOUT	87	88	87	88	87	88
STITLE	21	63	1	12	HEADER	7	7	32	32	32	32
SVALUE	21	75	1	6	EXTRA3	2	2	2	2	2	2
LTITLE	22	63	1	9	HEADER	10	10	32	32	32	32
LVALUE	22	72	1	9	HEADER	9	9	32	32	32	32
ENTRY	23	1	1	120	HEADER	4	4	10	10	32	32
VLINE	24	41	1	2	HEADER	5	5	10	10	32	32
PASSWD	24	43	1	38	SMOUT	45	45	45	45	46	46

QUERY COMPLETE

Figure 3.6 - Sample Output From the QUERY.WINDOWS Command

RESET

Use this command to restart your Session Manager display environment. This command will remove all entries from all of the stacks (see the SAVE and RESTORE commands) and reexecute the commands that create the IBM-supplied default screen layout. The TSOIN and TSOOUT streams will not be altered.

The RESET command should not be followed by other Session Manager commands on the same line. A command entered at the same time will execute before RESET processing can reestablish the default screen layout. Thus, any command entered with RESET will execute in an environment with no windows or Program Function keys defined.

Note: If you have accidentally deleted all the windows of your display environment, you can press the CLEAR key, then type RESET to get back the default screen layout.

```
RESET
```

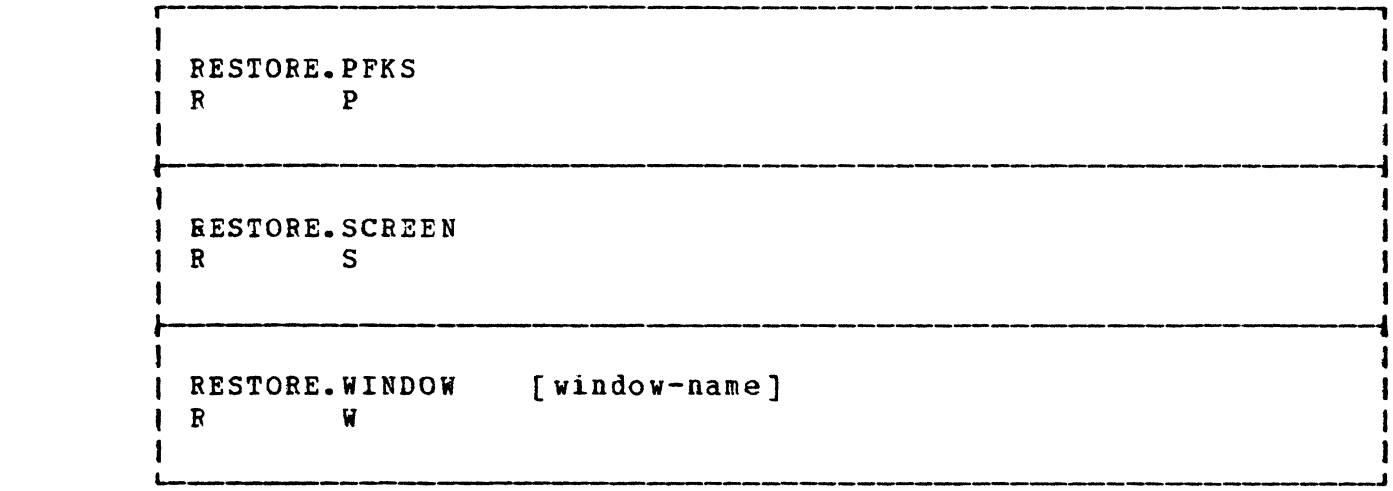
SAVE and RESTORE

The SAVE and RESTORE commands work together to allow you to more easily manipulate your display environment. The Session Manager maintains three push-down ("LIFO": Last In, First Out) stacks. These stacks are used to hold descriptions of portions of the display environment. By using the SAVE and RESTORE commands you can temporarily modify the screen layout, PF key definitions, or window attributes, and restore them to their previous status.

For example, you might SAVE the default PF key definitions, define the PF keys as TSO TEST subcommands for use with TSO TEST, then RESTORE their previous definitions when you leave TEST.

At the start of your session, the stacks are empty, issuing the RESTORE command has no effect at this time. Once an item has been saved on one of the stacks, the item remains as the first (bottom) item of the stack. It cannot be removed from the stack by issuing the RESTORE command. This allows you to set up your own "default" screen layout or PF key definitions: you can always recover to the first saved element on the stack by issuing the RESTORE command enough times to reach the bottom element (that is, as many times as you have saved elements above the first one).

Note: The RESET command removes all elements from the stacks (including the bottom elements).



PFKS (modifier)

specifies that the stack containing the definitions of the Program Function keys is to have its top element "popped": the top element is removed from the stack and used to restore the Program Function keys to the definitions they had at the time they were saved on the stack via the SAVE command.

Note: If the element is the only element on the stack (the bottom element), it is not removed from the stack.

SCREEN (modifier)

specifies that the stack containing the screen layout is to have its top element "popped": the top element is removed from the stack and used to restore the screen to the layout it had at the time it was saved on the stack via the SAVE command. Included in each SCREEN stack element is a description of the screen layout, the cursor location, the CONTROL setting (including the last non-zero control value entered - see the CHANGE.TERMINAL command), the name of the Default Window, and for each window, all the information described below under "WINDOW".

Note: If the element is the only element on the stack (the bottom element), it is not removed from the stack.

WINDOW (modifier)

specifies that the stack containing descriptions of window definitions is to have its top element "popped": the top element is removed from the stack and used to restore the specified window to the attributes of the win-

dow definition that was saved on the stack via the SAVE command. Each window description element in the stack contains the following information:

- The ALARM setting of the window
- The HOLD setting of the window
- The OVERLAP setting of the window
- Whether or not the window is protected
- The target stream of the window and the display intensity at which data is to be sent there
- The UPDATE mode of the window
- The stream the window is viewing
- The window's position on the stream it is viewing
- Whether the window is locked or unlocked

If the element is the only element on the stack (the bottom element), it is not removed from the stack.

Location and size information are not restored.

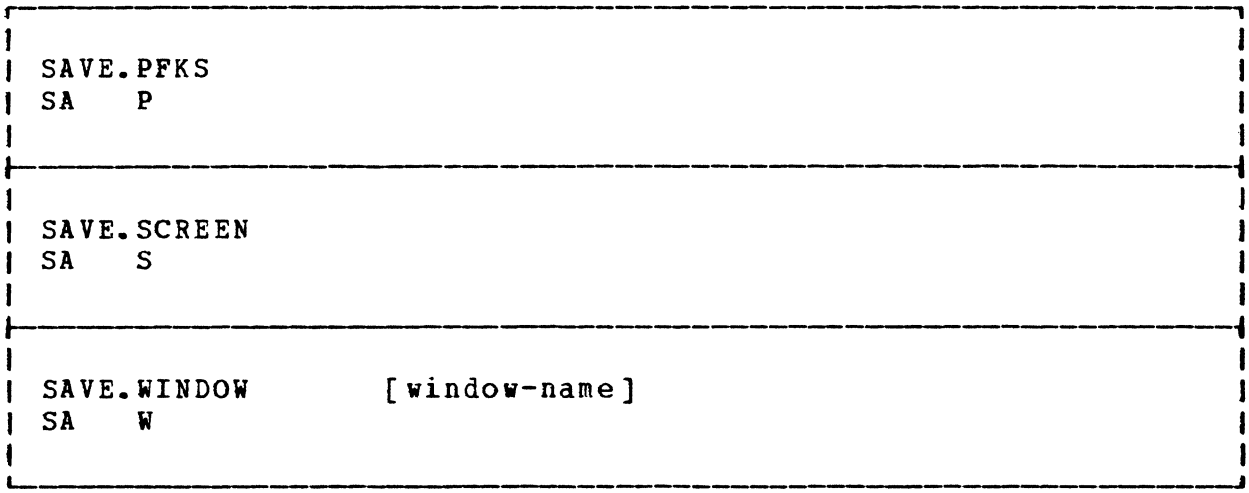
window-name (positional)

Default: the Default Window

is the name of the window whose description is to be restored. If no window name is specified, the top element of the window stack is popped from the stack and restored to the Default Window.

Example:

See the TSO CLISTs in "Using the Session Manager Commands" for examples of the use of the SAVE and RESTORE commands.



PFKS (modifier)

specifies that all the current PF key definitions are to be saved as the top element of the PF key stack. Any previously saved PF key definitions are "pushed down" on the stack.

SCREEN (modifier)

specifies that the current screen layout is to be saved as the top element of the SCREEN stack. Any previously saved screen layouts are "pushed down." Included in each SCREEN stack element is a description of the screen layout, the cursor location, the CONTROL setting (including the last non-zero control value entered - see the CHANGE.TERMINAL command), the name of the Default Window, and for each window, all the information described below under "WINDOW".

WINDOW (modifier)

specifies that the stack containing descriptions of window definitions is to be saved as the top element of the WINDOW stack. Each window description element in the stack contains the following information:

- The ALARM setting of the window
- The HOLD setting of the window
- The OVERLAP setting of the window
- Whether or not the window is protected
- The target stream of the window and the display intensity at which data is to be sent there
- The UPDATE mode of the window
- The stream the window is viewing
- The window's position on the stream it is viewing
- Whether the window is locked or unlocked

Location and size information are not saved.

window-name (positional)

Default: the Default Window

is the name of the window whose description is to be saved. If no window name is specified, the description of the "default window" is saved on the stack.

Example:

See the TSO CLISTs in "Using the Session Manager Commands" for examples of the use of the SAVE and RESTORE commands.

SNAPSHOT

This command takes a "snapshot" of the display screen and places it in the specified stream. The SMCOPY command (see "TSO Commands") can then be used to print the stream or copy it to a data set.

```
SNAPSHOT          target-stream-name
SN
                  [FORMAT]
```

target-stream-name (positional)

Default: none

is the name of the stream to which the snapshot is to be sent.

FORMAT (keyword)

Default: none
Alternate Form: FMT

specifies that carriage control information is to be included in the snapshot for printing on a system printer. Highlighted lines on the screen when the snapshot is made will appear darker in the printed copy.

Note: The PREFORMAT operand of the SMCOPY command must be used when printing the stream containing the snapshot if the FORMAT operand was entered with the SNAPSHOT command.

Example:

Record the image of the display screen in the EXTRA1 stream:

```
snapshot extra1 format
```

To print the stream, enter this TSO command:

```
smcopy fromstream(extra1) preformat
```

Sample output from the SNAPSHOT command is shown in Figure 3.7.

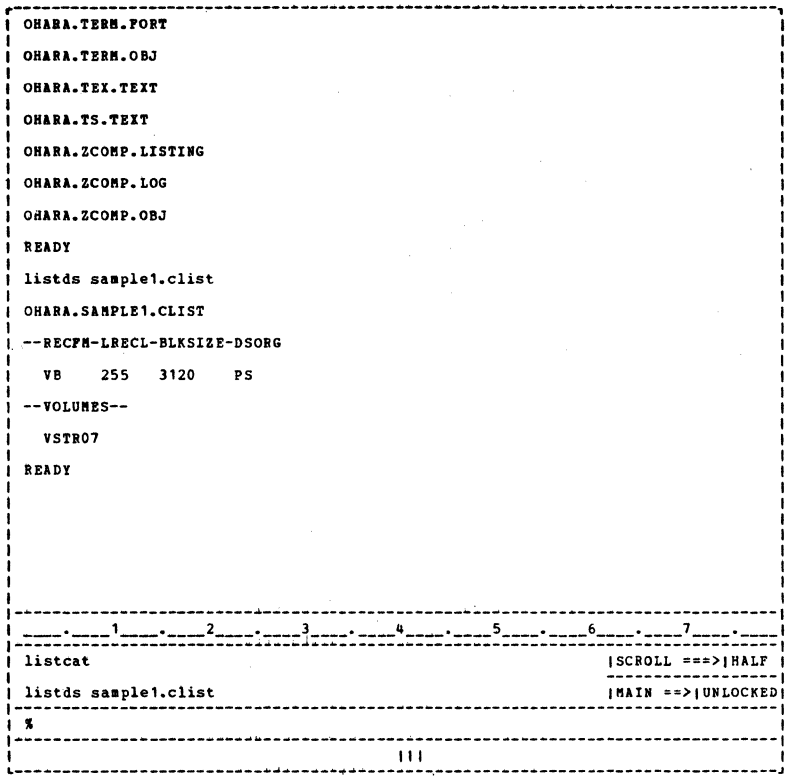


Figure 3.7 - Sample Output From the SNAPSHOT Command

TSO COMMANDS

Three TSO commands are provided for use with the Session Manager. They may be entered from the keyboard, from PF key definitions, or stored in and executed from CLISTs, just like any other TSO command.

Note: These commands are TSO Command Processors, not Session Manager commands.

See "OS/VS2 TSO Command Language Reference", GC28-0646, for syntax rules and data set naming conventions for these and all TSO commands.

SMCOPY

This command copies all or part of a stream or data set to another stream or data set (that is, stream to stream, stream to data set, data set to stream, data set to data set). The target data set may be a SYSOUT data set: Session Manager streams and TSO data sets may be printed at a system printer. Also, sequences of TSO commands (from the TSOIN stream), Session Manager commands (from the SMIN stream), TSO input and output (from the TSOOUT stream) may be copied to other streams (for reexecution, perhaps) or to data sets for printing, editing, and so on. Once saved in a data set, the contents may be copied back to a stream for display or reexecution.

Note: When the source and target of the copy are both data sets, (SYSOUT or QSAM), the user does not have to be logged on under the Session Manager to use the SMCOPY command.

SMCOPY returns these codes in Register 15 upon completion:

0 SMCOPY has successfully executed.

4 SMCOPY has sucessfully executed. The copy ended at the end of the source.

8 An error occurred during execution of SMCOPY. Possible causes:
- An incomplete or invalid command was entered and SMCOPY was unable to prompt.
- SMCOPY was unable to allocate the needed data sets.
- The data set(s) specified were of an invalid type.
- The stream name(s) specified were not found.

12 An internal error occurred. See your system programmer.

```
SMCOPY    [FROMDATASET(dsname) |
SMC        FROMSTREAM(stream-name) ]

          [PRINT(sysout-class) |
           TODATASET(dsname) |
           TOSTREAM(stream-name) ]

          [ASIS | CAPS | NOTRANS]

          [FORMAT | NOFORMAT| PREFORMAT]

          [LINE(start-line[:stop-line]) ]
```

Copy Source:
FROMDATASET | FROMSTREAM
Default: FROMSTREAM(TSOOUT)

Copy Target:
PRINT | TODATASET | TOSTREAM
Default: PRINT(A)

FROMDATASET(dsname) (keyword)

Default: none
Alternate Form: FDS

is the name of the data set to be used as the input source for the copy. It must be a sequential data set or a member of a partitioned data set with either fixed or variable length records.

This operand is mutually exclusive with FROMSTREAM.

FROMSTREAM(stream-name) (keyword)

Default: none
Alternate Form: FS

is the name of the input stream for the copy operation.

This operand is mutually exclusive with FROMDATASET.

PRINT(sysout-class) (keyword)

Default: none

specifies that the input source is to be copied to a sysout data set of the specified sysout class and print-

ed on a system printer.

This operand is mutually exclusive with TODATASET and TOSTREAM.

TODATASET(dsname) (keyword)

Default: none
Alternate Form: TDS

is the name of the data set to be used as the output target for the copy. It must be a sequential data set or a member of a partitioned data set with either fixed or variable length records.

If the data set does not exist, a new one will be allocated. If the source for the copy is a data set (FROMDATASET or FROMDD is specified), its attributes will be used to allocate the target data set. If the source for the copy is a stream (FROMSTREAM is specified), the data set will be allocated with these attributes:

RECFM VB. VBA if FORMAT or PREFORMAT is specified (see below).

LRECL 255

BLKSIZE 3120

This operand is mutually exclusive with PRINT, and TOSTREAM.

TOSTREAM(stream-name) (keyword)

Default: none
Alternate Form: TS

is the name of the output stream for the copy operation.

This operand is mutually exclusive with PRINT and TODATASET.

ASIS | CAPS | NOTRANS (keyword)

Default: CAPS

specifies what character translation, if any, is to be performed during the copy operation:

ASIS specifies that unprintable characters in the input source will be translated to blanks (hex '40'). Lower case characters in the source will be left as lower case.

ASIS is used if the source is to be printed on a printer with a dual-case print train (TN or T11).

CAPS specifies that unprintable characters in the input source will be translated to blanks (hex '40'). Lower case characters in the source will be translated to upper case.

CAPS is used if the source is to be printed on a printer with a single-case print train (PN or P11).

NOTRANS specifies that no translation is to occur.

FORMAT | NOFORMAT | PREFORMAT (keyword)

Default: NOFORMAT
Alternate Forms: FMT | NOFMT | PREFMT

specifies whether carriage control characters are to be placed with the data in the output target:

FORMAT specifies that carriage control characters are to be placed in the output target such that highlighted lines in an input stream will be overprinted.

If the output target is a stream, highlighted lines in the source will appear highlighted in the target copy.

If the output target is a data set, the record format must be FBA or VBA to indicate the presence of ASA control characters. If the target data set is new, it will be allocated with a VBA record format.

Note: This operand is ignored if FROMSTREAM is not specified as the input source.

NOFORMAT specifies that no control characters are to be placed in the output target.

If the source is a data set, it must have a FB or VB record format. If the source is a stream and the target is a data set, the data set must have a FB or VB record format. If both the source and target are data sets, they must have the same record format (FB or VB).

If the target data set is new, it will be allocated with a VB record format (if the source is a stream) or it will be allocated with the same

record format as the source data set (for data set to data set copy).

PREFORMAT
specifies that the source for the copy (stream or data set) already contains carriage control characters. This operand is used when the source stream contains output from the SNAPSHOT command.

If the source is a data set, it must have a FBA or VBA record format. If the source is a stream and the target is a data set, the data set must have a FBA or VBA record format. If both the source and target are data sets, they must have the same record format (FBA or VBA).

If the target data set is new, it will be allocated with a VBA record format (if the source is a stream) or it will be allocated with the same record format as the source data set (for a data set to data set copy).

LINE(start-line:stop-line) (keyword)

Default: start-line - first line of source
stop-line - last line of source

specifies the range of lines to be copied.

If the source is a stream, specific stream-line numbers may be found with the QUERY command or the SMFIND command.

If the source is a data set, these line numbers represent records of the data set, not the line number of numbered data sets.

Example:

Copy the TSOOUT stream to the system printer, translating all lower case letters to upper case:

```
smcopy
```

Copy the EXTRA1 stream to the system printer, having previously used the SNAPSHOT command to place an image of the display screen there:

```
smcopy fromstream(extra1) print(t) preformat
```

Note: Here, SYSOUT Class T will direct the output to a printer with a dual-case print train. Check with your system pro-

grammer to determine the proper SYSOUT class to specify.

Example:

Copy a member of the data set 'SYS1.CLIST' to a data set BRUCE.CLIST:

```
smc fds('sys1.clist(zlogon)') tds(bruce.clist(zlogon))
```

Note: You do not need to be logged on under the Session Manager to perform data set to data set copies.

Example:

Copy a data set containing TSO commands to the TSOIN stream, where they will be executed:

```
smcopy fds('linda.commands.data') ts(tsoin)
```

SMFIND

Use this command to locate a string of characters in a Session Manager stream. If the text string is found, its line number is displayed in the TSOOUT stream and returned in Register 15. TSO CLISTs can then access the line number from the CLIST variable "&LASTCC."

The SMFIND command is most useful when used in conjunction with other Session Manager commands and TSO CLISTs.

SMFIND returns these codes in Register 15 upon completion:

line	A positive integer specifying in which line of the stream the text was found. The maximum value is 16777216, limited by the &LASTCC variable, not by SMFIND.
0	The text was not found, or the stream specified does not exist, or the command was incorrectly specified and SMFIND was unable to prompt for correct information.

```
SMFIND  text-string
SMF
       [STREAM(stream-name)]

       [BACKWARD | FORWARD]

       [ALL | FIRST]

       [ANY | ASIS]

       [LINE(line1[:line2])]
```

text-string (positional)

Default: none

is the character string to be searched for in the specified stream. It must be no more than 256 characters in length. It must be enclosed in delimiters (characters) not present in the text-string.

STREAM(stream-name) (keyword)

Default: TSOOUT

is the name of the stream to be searched.

BACKWARD | FORWARD (keyword)

Default: BACKWARD

specifies the direction of the search within the range of line numbers (see LINE below).

ALL | FIRST (keyword)

Default: FIRST

specifies whether all occurrences or just the first occurrence of the text-string are to be found.

ALL specifies that all occurrences of the text-string are to be found. The line number of each line in the stream that contains the text-string will be displayed in the TSOOUT stream, and Register 15 (and the CLIST variable "&LASTCC") will contain the line number of the last occurrence of the text-string.

FIRST specifies that only the first occurrence of the

text-string is to be found. The line number of the line in the stream that contains the text-string will be displayed in the TSOOUT stream, and Register 15 (and the CLIST variable "&LASTCC") will contain that line number.

ANY | ASIS (keyword)

Default: ASIS

specifies whether the found text is to be an exact match of the text-string. If ANY is specified, upper and lower case differences will be ignored during the search.

LINE(line1:line2) (keyword)

Default: see below

specifies the range of lines to be searched. If LINE is not specified, the entire stream is searched. The direction of the search within the range of lines is controlled by BACKWARD or FORWARD.

If only "line1" is specified, the search will proceed from that line to the top or bottom of the stream, depending on whether BACKWARD or FORWARD was specified.

If either "line1" or "line2" is not in the stream, the closest line in the stream will be used instead.

Example:

Find the first occurrence of 'time' in the TSOOUT stream:

```
smfind /time/ forward
```

Example:

Find the last occurrence of 'time' in the TSOOUT stream:

```
smfind ,time,
```

Example:

Use SMCOPY in conjunction with SMFIND to save a record of a just completed TSO TEST session in a data set called TESTLOG.TEXT:

```
smfind /test/
```

This will display the line number on which you entered the TEST command on begin your TEST session. Assume it was line 293.

```
smfind /ready/ for any 1(293)
```

This will find the first READY message after the TEST command, that is when you ended the test session. Assume it was at line 436.

```
smcopy toda(testlog.text) li(293:436)
```

This will copy just lines 293 to 436 to the data set.

Example:

Use a CLIST to print the results from the execution of an interactive program. The CLIST would be:

```
SMFIND /CALL/ ANY
SET STARTLIN = &LASTCC
SMFIND /READY/ FORWARD LINE(&STARTLIN.)
SET ENDLIN = &LASTCC
SMCOPY FMT LINE(&STARTLIN.:&ENDLIN.)
```

Example:

With the default window at the bottom of the stream it is viewing, scroll it so the last time you typed "listcat" is at the top of the window:

```
smfind /listcat/
```

Assume the text was found at line 525. Now, from TSO, cause the window to scroll to that line:

```
smput /scroll absolute 525/
```

Note: You could do this same function using the Session Manager commands FIND and SCROLL.

SMPUT

This command sends the specified text string to the specified Session Manager stream.

If the stream specified is SMIN or TSOIN the text string will be interpreted as a command and executed by the Session Manager or TSO, respectively. Since this command is a TSO Command Processor, it can be stored in and executed from TSO CLISTs. By defining the text string as a Session Manager command and directing it to the SMIN stream, you could store the necessary Session Manager commands to define various environments (for command entry, editing, program execution, and so on) in CLISTs to dynamically change your environment during the terminal session.

SMPUT returns these codes in Register 15 after execution:

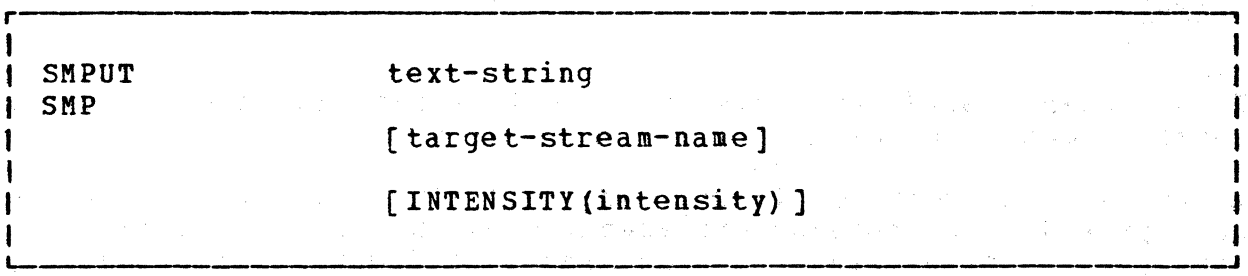
0 SMPUT has successfully executed.

Note: If the text string contained Session Manager or TSO commands, the zero return code does not indicate successful execution of those commands.

4 An error occurred while placing the text string in the stream.

8 The target stream was not found. The Session Manager may not be active.

12 An error occurred while parsing the entered command.



text-string (positional)

Default: none

is the character string to be sent to the specified stream, enclosed in delimiters (characters) not present in the text-string. It must be no longer than 32768 characters, excluding the delimiters.

Note: If the target stream is the SMIN stream, the text-string should be no longer than 512 characters. The Session Manager command interpreter truncates input lines at 512 characters.

target-stream-name (positional)

Default: SMIN

is the name of the stream to which the text string is to be sent.

Note: This operand is required if the INTENSITY operand is specified.

INTENSITY(intensity) (keyword)

Default: 1

is the brightness at which the data in the stream is to be displayed. Valid values are:

0 The data is not to be displayed.

1 The data is to display at normal intensity.

2 The data is to be highlighted.

Example:

See "Using the Session Manager Commands" for examples of SMPUT.

4. THE DEFAULT DISPLAY ENVIRONMENT

As you begin your Session Manager/TSO session, an IBM-supplied default display environment is established for you. This environment includes streams, session functions, a screen layout comprised of multiple windows, and PF key definitions. You can change these definitions with the various Session Manager commands to suit your particular needs.

The IBM-supplied default screen layout, as shown in Figure 1.3, is not rigidly built into the programs comprising the Session Manager. During Session Manager initialization, a list of Session Manager commands are executed to create the screen layout. This section lists the commands used to create the IBM-supplied default screen layout, and explains how they work together with the streams to create a useable terminal environment.

Note: The use of the default screen layout and the default Program Function key definitions is described in "A Session Manager Primer".

Default Streams

There are nine streams created when you begin your session. You cannot change these streams except as provided in the CHANGE.STREAM command. Your installation, however, can change the streams as described in "System Programmer Reference". The names and attributes of the IBM-supplied streams are as follows:

STREAM NAME	SIZE LINES	BYTES	TYPE	ALARM
TSOIN	305	8192	INPUT	NO
TSOOUT	4005	147456	OUTPUT	NO
EXTRA1	405	32768	OUTPUT	NO
SMOUT	155	4096	OUTPUT	NO
HEADER	55	1024	EXTRA	NO
EXTRA3	105	1024	EXTRA	NO
EXTRA2	105	1024	EXTRA	NO
MESSAGE	55	1024	OUTPUT	YES
SMIN	305	8192	INPUT	NO

Default Screen Layout and PF Key Definitions

The layout of the default screen and the definitions of the Program Function keys are described in "A Session Manager Primer". Below are listed the commands used to create the default screen layout and some comments explaining how they work together:

Note: Long commands are shown continued on the next line, indented three spaces.

The first group of commands places data in the HEADER and EXTRA3 streams. This sets up title lines for the windows used as visual dividers to view. The stream is first cleared so that when these default commands are re-executed (when you enter the RESET command) the data are always placed in the same line of the stream.

```
CHANGE.STREAM HEADER CLEAR
PUT '*** THIS IS THE HEADER STREAM ***' HEADER
PUT '____.____1____.____2____.____3____.____4____.____5____.____6
   ____.____7____.____8____.____9____.____10___.____11___.____12_
   __.____13___.____14___.____15___.____16___.____17___.____18___
   .____19___.____20___.____21___.____22___.____230' HEADER INTEN
   SITY(2)
PUT '%' HEADER INTENSITY(2)
PUT '|' HEADER INTENSITY(2)
PUT ' ' HEADER INTENSITY(0)
PUT 'SCROLL ===>' HEADER INTENSITY(2)
PUT 'UNLOCKED' HEADER INTENSITY(2)
PUT '  LOCKED' HEADER INTENSITY(2)
PUT 'MAIN ==>' HEADER INTENSITY(2)
PUT 'HALF' EXTRA3 INTENSITY(2)
```

The windows comprising the screen layout are defined next.

The MAIN and CURRENT windows were described in "A Session Manager Primer". The LINE window views the numbered, dashed line in the HEADER stream to provide visual separation between the MAIN and CURRENT windows.

```
DEFINE.WINDOW MAIN 1 1 19 80 ALARM(NO) HOLD(1) OVERLAP(9) PROTECT
   (NO) TARGET(TSOIN 1) UPDATE(LINE) VIEW(TSOOUT)
DEFINE.WINDOW LINE 20 1 1 80 ALARM(NO) HOLD(0) OVERLAP(0) PROTECT
   (YES) UPDATE(NEWEST) VIEW(HEADER)
SCROLL.ABSOLUTE 2 LINE
DEFINE.WINDOW CURRENT 21 1 2 62 ALARM(NO) HOLD(0) OVERLAP(0) PROT
   ECT(NO) TARGET(TSOIN 1) UPDATE(NEWEST) VIEW(TSOOUT)
```

The STITLE and LTITLE windows both view the HEADER stream, displaying "SCROLL ===>" and "MAIN ==>", respectively.

The SVALUE window views the EXTRA3 stream, displaying the current scroll amount value. This value is placed in the EXTRA3 stream when PF 2 is pressed (see below).

The LVALUE window views the HEADER stream, displaying "LOCKED" or "UNLOCKED". The PF key definitions move this window to line 7 or 8 of the HEADER stream to display the appropriate message.

```
DEFINE.WINDOW STITLE 21 63 1 12 ALARM(NO) HOLD(0) OVERLAP(0) PROT
   ECT(YES) UPDATE(NEWEST) VIEW(HEADER)
SCROLL.ABSOLUTE 6 STITLE
DEFINE.WINDOW SVALUE 21 75 1 6 ALARM(NO) HOLD(0) OVERLAP(0) PROTE
   CT(YES) TARGET(EXTRA3) UPDATE(NEWEST) VIEW(EXTRA3)
DEFINE.WINDOW LTITLE 22 63 1 9 ALARM(NO) HOLD(0) OVERLAP(0) PROTE
   CT(YES) UPDATE(NEWEST) VIEW(HEADER)
SCROLL.ABSOLUTE 9 LTITLE
DEFINE.WINDOW LVALUE 22 72 1 9 ALARM(NO) HOLD(0) OVERLAP(0) PROTE
   CT(YES) UPDATE(NEWEST) VIEW(HEADER)
SCROLL.ABSOLUTE 7 LVALUE
```

The ENTRY window serves as the primary entry area on the screen. The permanent cursor is in this window. The window views the HEADER stream, displaying a "%" to make implicit executing of CLISTs easier. Notice that this window wraps to the 24th line of the display screen as it is 120 characters wide.

The VLINE window serves as a visual divider between the ENTRY and PASSWD windows. It displays a "|" in the HEADER stream.

The PASSWORD window serves two purposes. First, it is used as an entry area for TSO passwords (the target stream is TSOIN, and data sent there does not display). Second, Session Manager error messages are displayed in this window (it is viewing the SMOUT stream).

```
DEFINE.WINDOW ENTRY 23 1 1 120 ALARM(NO) HOLD(0) OVERLAP(0) PROTE
   CT(NO) TARGET(TSOIN 1) UPDATE(NEWEST) VIEW(HEADER)
SCROLL.ABSOLUTE 3 ENTRY
DEFINE.WINDOW VLINE 24 41 1 2 ALARM(NO) HOLD(0) OVERLAP(0) PROTEC
   T(YES) UPDATE(NEWEST) VIEW(HEADER)
SCROLL.ABSOLUTE 4 VLINE
DEFINE.WINDOW PASSWD 24 43 1 38 ALARM(NO) HOLD(0) OVERLAP(0) PROT
   ECT(NO) TARGET(TSOIN 0) UPDATE(NEWEST) VIEW(SMOUT)
```

The Program Function keys are defined next.

PF 1 issues the SMCOPY TSO command to print the contents of the EXTRA1 stream (where PF 4 places snapshots of the screen). The SMPUT TSO command is then used to issue the CHANGE.STREAM command to erase the contents of the EXTRA1 stream. The SMPUT command is used to send the CHANGE.STREAM command to the Session Manager to insure the stream is not cleared before the SMCOPY command executes.

```
CHANGE.PF 1 'PUT ''SMCOPY FROMSTREAM(EXTRA1) PRINT(A) PREFORMAT C
   APS'' TSOIN;PUT ''SMPUT /CHANGE.STREAM EXTRA1 CLEAR/'' TSOIN'
   SMIN
```

PF 2 accepts as input a scroll amount value. This value is placed in the EXTRA3 stream where the SVALUE window will display it, and then each of the four scrolling PF keys is redefined to use this new value. Each PF key is defined as several Session Manager commands:

- The Default Window is first scrolled 0 lines or columns. This is done to insure that when the PF key is pressed, the window is locked, even if the amount to scroll is invalid (see below).
- The Default Window is scrolled the entered amount. If this is an invalid value (such as "xxx") this command will cause an error message to be displayed.
- The CONTROL timer is set to 0. This causes the keyboard to remain unlocked so additional scrolling keys may be pressed.
- The LVALUE window is scrolled to line 8 of the HEADER stream, displaying "LOCKED".

```
CHANGE.PF 2 'PUT ''&1.'' EXTRA3 INTENSITY(2);CHANGE.PF 7 ''SCROLL
   .BACKWARD 0;SCROLL.BACKWARD AMOUNT(&1.);CHANGE.TERMINAL CONTRO
   L(0);SCROLL.ABSOLUTE 8 LVALUE'' SMIN;CHANGE.PF 8 ''SCROLL.FORW
   ARD 0;SCROLL.FORWARD AMOUNT(&1.);CHANGE.TERMINAL CONTROL(0);SC
   ROLL.ABSOLUTE 8 LVALUE'' SMIN;CHANGE.PF 10 ''SCROLL.LEFT 0;SCR
   OLL.LEFT AMOUNT(&1.);SCROLL.LEFT LINE AMOUNT(&1.);SCROLL.ABSOL
     UTE 8 LVALUE;CHANGE.TERMINAL CONTROL(0)'' SMIN;CHANGE.PF 11
   ''SMIN;CHANGE.PF 11 ''SCROLL.RIGHT 0;SCROLL.RIGHT AMOUNT(&1.);
    SCROLL.RIGHT LINE AMOUNT(&1.);CHANGE.TERMINAL CONTROL(0); SCR
   OLL.ABSOLUTE 8 LVALUE'' SMIN' SMIN SUBSTITUE
```

PF 3 sets up the screen and PF keys for entry of Session Manager commands, as described in "Using the Session Manager Commands".

PF 4 takes a snapshot of the screen and places it in the EXTRA1 stream. It also places a message in the SMOUT stream.

PF 5 issues the FIND.BACKWARD command. It accepts as input the search argument for the command. Notice the use of quotes so that you don't need to enclose the search argument in quotes.

```
C.P 3  'C.F SM C(SMOUT 1);SAVE.PF;C.W PASSWD V(HEADER);S.A 5 PASS
   WD;PUT '' '' SMOUT;PUT '' '' SMOUT;$*.;C.W CURRENT T(SMIN) V(S
   MOUT);C PF 3 ''&*.'' SMIN S;C.PF 6 ''C.F SM C(SMOUT 0);PUT
   '''' '''' SMOUT;C.W PASSWD V(SMOUT);R PF;CHANGE.WINDOW CU
   RRENT TARGET(TSOIN) VIEW(TSOOUT) '' SMIN' SMIN SUB($)
CHANGE.PF 4 'SNAPSHOT EXTRA1 FORMAT;PUT ''SNAPSHOT COMPLETE'' SMO
   UT INTENSITY(2)'' SMIN
CHANGE.PF 5 'SCROLL.BACKWARD 0;FIND.BACKWARD ''&*.'';CHANGE.TERMI
   NAL CONTROL(0);SCROLL.ABSOLUTE 8 LVALUE' SMIN SUBSTITUTE
```

PF 6 uses the SAVE and RESTORE commands to redefine itself to act as a flip-flop key, causing the CURRENT window to view the TSOOUT

stream or the TSOIN stream.

PF 7 - PF 11 are the scrolling PF keys. Notice that each key, in addition to scrolling the Default Window, sets the CONTROL timer to 0 and moves the LVALUE window to indicate the locked status of the Default Window.

PF 12 unlocks the Default window, resets the CONTROL timer to its last non-zero value and moves the LVALUE window to indicate the unlocked status of the Default Window.

```
CHANGE.PF 6 'SAVE.PF;CHANGE.WINDOW CURRENT TARGET(TSOIN 1) VIEW(T
   SOIN);CHANGE.PF 6 ''CHANGE.WINDOW CURRENT TARGET(TSOIN 1) VIEW
   (TSOOUT);RESTORE.PF'' SMIN' SMIN
CHANGE.PF 7 'SCROLL.BACKWARD AMOUNT(HALF);CHANGE.TERMINAL CONTROL
   (0);SCROLL.ABSOLUTE 8 LVALUE' SMIN
CHANGE.PF 8 'SCROLL.FORWARD AMOUNT(HALF);CHANGE.TERMINAL CONTROL(
   0);SCROLL.ABSOLUTE 8 LVALUE' SMIN
CHANGE.PF 9 'SCROLL.OLDEST;CHANGE.TERMINAL CNTL(0);SCROLL.ABSOLU
   TE 8 LVALUE' SMIN
CHANGE.PF 10 'SCROLL.LEFT AMOUNT(HALF);SCROLL.LEFT LINE AMOUNT(HA
   LF);CHANGE.TERMINAL CONTROL (0);SCROLL.ABSOLUTE 8 LVALUE' SMIN
CHANGE.PF 11 'SCROLL.RIGHT AMOUNT(HALF)SCROLL.RIGHT LINE AMOUNT (
  HALF);CHANGE.TERMINAL CONTROL(0);SCROLL.ABSOLUTE 8 LVALUE' SMIN
CHANGE.PF 12 'UNLOCK NEWEST;CHANGE.TERMINAL CONTROL(LAST);SCROLL.
   ABSOLUTE 7 LVALUE' SMIN
```

The PF key definitions for PF 13-24 are the same as PF 1-12. Once the PF keys are defined, other Session Manager defaults are set up.

The CONTROL timer is set to 15, and the MAIN window is named the Default Window.

The cursor is placed in the ENTRY window.

The session functions are set up. Notice that the Session Manager copies its input to the SMOUT stream at 0 intensity. This causes an invisible line to be displayed in the PASSWD window (see above), thus when passwords are entered in the PASSWD window, they are not displayed as they are typed.

```
CHANGE.TERMINAL CONTROL(15) DEFAULT(MAIN)
CHANGE.CURSOR 1 1 ENTRY
CHANGE.FUNCTION MSG OUTPUT(TSOOUT 2) ALARM(OUTPUT)
CHANGE.FUNCTION SM INPUT(SMIN) COPY(SMOUT 0) OUTPUT(SMOUT 2) ALAR
   M(OUTPUT)
CHANGE.FUNCTION TSO INPUT(TSOIN) COPY(TSOOUT 2) OUTPUT(TSOOUT 1)
   ALARM(NO)
CHANGE.STREAM SMIN ALARM(NO)
CHANGE.STREAM SMOUT ALARM(NO)
CHANGE.STREAM TSOIN ALARM(NO)
CHANGE.STREAM TSOOUT ALARM(NO)
CHANGE.STREAM HEADER ALARM(NO)
```

```
CHANGE.STREAM EXTRA1 ALARM(NO)
CHANGE.STREAM EXTRA2 ALARM(NO)
CHANGE.STREAM EXTRA3 ALARM(NO)
```

5. SYSTEM PROGRAMMER REFERENCE

This section describes the changes necessary to get the Session Manager environment defined properly. Included are details regarding changes to the TSO/TCAM message handler, TSO parameter changes and optional changes to the Session Manager stream definitions.

TSO Parameter Changes

In order to enable use of the Session Manager, either new LOGON Procedures must be created or existing LOGON Procedures must have EXEC statement modifications. The EXEC statement may have the following format:

```
// EXEC PGM=ADFMDF03,
// PARM='SM(tmpname,Y|N,default-module-name),normal TMP parms'
```

The program name ADFMDF03 causes the Session Manager Initialization task to be attached instead of the TMP. After initialization has completed, the TMP will be attached using the TMP name found in the Session Manager parameters in the PARM field. In addition, APF authorization requirements (Y or N) and the name of the Session Manager default environment module are gotten from the PARM field.

Parameters needed to be passed to the TMP may be specified after the Session Manager Parameters.

Certain defaults apply and will be described after the syntax.

```
[PARM='[SM(tmpname,Y|N,default-module-name)],[tmp-parms]']
```

Each of the parameters in the SM subfield are positional, so commas must be used as place-holders when allowing either of the first two parameters to default. For example, SM(,,default-module-name).

As shown above, the entire PARM field is optional. If nothing is supplied, the default TMP module name invoked is IKJEFT01 and it is APF authorized. If the PARM field only specified SM(IKJEFT01), then it is APF authorized. If no default environment module name is specified, then the IBM supplied default ADFMDFLT is used.

In addition, the following changes must be made:

- The new data set SYS1.SMLIB must be added to the APF authorization list.
- The Session Manager module ADFMDF03 must be included in the system pageable link pack area.
- The link list member LNKLSTxx of SYS1.PARMLIB must be updated to include SYS1.SMLIB.

TSO/TCAM Message Handler Changes

Note: These changes should be bypassed if SPF is installed or the Session Manager is to run under control of MVS/TSO/VTAM.

Minor modifications must be made to the standard TSO/TCAM message handler, for the operation of the Session Manager. Creation of the message handler normally requires two assemblies (stage 1 and stage 2), where the output from the first assembly becomes the input to the second.

STAGE 1 MODIFICATION

If the full screen (FULLSCR=YES) operand was not originally specified during message control program (MCP) generation, the following change should be made and stage 1 should be reassembled. Alternatively, procedure 1b below may be followed during stage 2 modifications.

Add the FULLSCR=YES operand to the LINEGRP (and/or LISTTA) macros which describe the terminals being used and reassemble the MCP generation deck. This will result in the generation of the TCAM MCP source deck with:

- the generation of the OPTION macro for IEDQFSCR,
- the allocation and initialization (to zero) of IEDQFSCR in the appropriate TERMINAL macros via the OPDATA operand, and
- the generation of the FULLSCR macro in the TSO message handler, immediately preceding the SIMATTN macro.

STAGE 2 MODIFICATIONS

1a. If the FULLSCR=YES operand was specified during stage 1, make the following two modifications to the stage 1 output.

- Replace the FULLSCR macro statement in the TSO message

handler with the macro statement SPFSCRN (no operands). The FULLSCR macro may have had a label, which should be retained on the SPFSCRN macro.

- Insert the macro statement SPFMCHK (no operands) in the output side of the TSO message handler between the macro statements OUTBUF and CODE.

1b. If the FULLSCR=YES operand was not specified during stage 1, make the following modifications to the stage 1 output:

- Insert the macro statement SPFSCRN (no operands) in the TSO message handler immediately preceding the SIMATTN macro.

- Insert the macro statement SPFMCHK (no operands) in the output side of the TSO message handler between the macro statements OUTBUF and CODE.

- Insert the following OPTION statement at an appropriate place in the option table.

```
IEDQFSCR OPTION XL1
```

- Ensure that every TERMINAL macro representing a 3277 or 3275 to be used with the Session Manager has an OPDATA operand with a subfield of zero corresponding to the IEDQFSCR option.

Note: The subfields of the opdata operand are "positional" and correspond in order to the order of the OPTION macros that make up the option table.

- Ensure that every other TERMINAL macro with an OPDATA operand also has a null subfield of that operand corresponding to the IEDQFSCR option.

2. Before proceeding with the stage 2 assembly, examine the current operand values for BUFSIZE, CUTOFF, and LNUNITS and adjust as follows:

- Ensure that the value of the BUFSIZE operand is a least 2100 on both TERMINAL and DCB macros for all the Session Manager terminals. BUFSIZE should also be an even multiple of the KEYLEN operand value on the INTRO macro.

- Ensure that the operand value of the CUTOFF macro is at least 2048 (decimal) in the TSO message handler.

- Ensure that the value of the LNUNITS operand of the INTRO macro is large enough. It should be at least equal to the

$$\frac{(B)\quad (T)}{K} \quad \text{for all TCAM DCB's}$$

WHERE: B is the BUFSIZE value for a DCB macro,
T is the number of TERMINAL macros for that DCB,
K is the KEYLEN value on the INTRO macro.

When the above modifications have been made, stage 2 should be reassembled. The macros SPFSCRN and SPFMCHK will reside in SYS1.MACLIB. The new MCP should then be link edited and placed in SYS1.LINKLIB.

TSO System Parameters

The TSO system patameters should be reviewed for adequacy. For MVS/TSO/VTAM, check PARMLIB member TSOKEY00. The installation defaults for MVS/TSO/VTAM include the following:

Screen size: 480

Buffer size: 132

The screen size should be increased to 1920. The buffer size is not critical, but a larger value should improve system performance (recommended value: 1000 or greater). The other MVS/TSO/VTAM defaults should be adequate.

For TSO/TCAM, the TIOC buffer control parameters may be critical. The following algorithms should provide adequate buffer control performance:

- TIOC BUFSIZE = 2(TCAM KEYLEN) but not greater than 252

- $\text{TIOC OWAITHI} = \dfrac{1.2(t)\quad (M)}{(\text{TIOC BUFSIZE})}$

- TIOC BUFFERS = 1.2(t) (TIOC OWAITHI)

- $\text{TIOC RESVBUF} = \dfrac{\text{TIOC BUFFERS}}{\phantom{\text{TIOC BUFFERS}}}$

7

- TIOC OWAITLO = $\frac{\text{TIOC OWAITHI}}{4}$

- TIOC INLOCKHI = TIOC OWAITHI

- TIOC INLOCKLO = TIOC OWAITLO

where:

M is the maximum value specified on any TCAM BUFSIZE parameter in a DCB or TERMINAL macro used for TSO;

t is the total number to TCAM TERMINAL macros used for TSO.

When computations result in remainders, round-up.

Note: The TIOC OWAITHI value must be large enough to allow a full-screen write before swapout. The OWAITHI value times the BUFSIZE should not be less than 1920 (decimal), otherwise severe performance degradation may result.

Session Manager Default Environment

When being initialized, the Session Manager loads the default environment module which is specified as a parameter on the EXEC statement of the logon procedure. IBM supplies a defaults module, ADFMDFLT, that is loaded if the module name is not specified on the EXEC statement. The default environment module contains the tables and data necessary for building the TSO user's default screen layout, PF key definitions, etc. The IBM supplied defaults module provides an environment for the 3270 Model 2 display terminals (i.e., 24 lines by 80 columns) and twenty-four PF key definitions.

You may want to use one default environment for all your TSO users. In this case, you may simply modify the IBM supplied defaults module to refine the environment for your particular requirements. You may find that one environment does not satisfy the requirements of all your TSO users. In this case, you can create multiple logon procedures, each one specifying a different default environment module.

The next few sections deal with the various options open to you for modifying a default environment and/or providing multiple default environments to your TSO users.

Use of CLISTs to Modify a Default Environment

Assume that you are using the IBM supplied defaults module ADFMDFLT and that you want to make several modifications to the screen layout and PF key definitions. The easiest way, but least efficient, is to supply an EXEC TSO command as a TMP parameter on the EXEC statement of the logon procedure. This command could execute an installation CLIST that issues Session Manager commands to modify the defaults. This CLIST could also execute another CLIST, whose name is prefixed by the TSO user's userid, to provide a mechanism for the end user to further modify his or her own environment.

There are two main disadvantages to modifying the defaults this way:

- The only changes that can be made are those done by Session Manager commands. Stream sizes cannot be modified this way.

- If the end user issues the Session Manager RESET command, then the default environment is set up without the CLIST being executed again.

Use of AMASPZAP to Modify the Defaults Module

Assume that you are using the IBM supplied defaults module ADFMDFLT and that you want to modify some of the streams to make them larger or smaller. AMASPZAP can be used to modify ADFMDFLT as long as nothing is done to change the size of the module.

The next section describes the stream definitions provided in ADFMDFLT and the section after that describes what you can and cannot change in those definitions.

The disadvantages to simply altering ADFMDFLT are:

- There is very little you can change without changing the size of the module. The screen layout and PF key definitions would be very difficult to modify.
- Any change you make will affect any and all TSO users who are using the logon procedure(s) that causes ADFMDFLT to be loaded as the defaults module.
- You cannot provide different environments for different end users this way.

If you are primarily interested in creating multiple default environments, then you still should read the next two sections, because they describe the basic concepts for choosing certain stream attributes.

Default Stream Definitions

The default stream definitions are as follows:

```
Stream name: TSOIN
Stream size: 8192 bytes
Maximum lines: 300
Lines per IDB: 5
Stream type: 1
Stream header: *** THIS IS THE TSOIN STREAM ***
Header length: 32
Stream flags: 00000000
```

```
Stream name: TSOOUT
Stream size: 147456 bytes
Maximum lines: 4000
Lines per IDB: 5
Stream type: 2
Stream header: *** THIS IS THE TSOOUT STREAM ***
Header length: 33
Stream flags: 00000000

Stream name: SMIN
Stream size: 8192 bytes
Maximum lines: 300
Lines per IDB: 5
Stream type: 1
Stream header: *** THIS IS THE SMIN  STREAM ***
Header length: 32
Stream flags: 00000000

Stream name: SMOUT
Stream size: 4096 bytes
Maximum lines: 150
Lines per IDB: 5
Stream type: 2
Stream header: *** THIS IS THE SMOUT  STREAM ***
Header length: 33
Stream flags: 00000000

Stream name: HEADER
Stream size: 1024 bytes
Maximum lines: 50
Lines per IDB: 5
Stream type: 0
Stream header: *** THIS IS THE HEADER STREAM ***
Header length: 33
Stream flags: 80000000

Stream name: MESSAGE
Stream size: 1024 bytes
Maximum lines: 50
Lines per IDB: 5
Stream type: 2
Stream header: *** THIS IS THE MESSAGE STREAM ***
Header length: 34
Stream flags: 40000000

Stream name: EXTRA1
Stream size: 32768 bytes
Maximum lines: 400
Lines per IDB: 5
Stream type: 2
Stream header: *** THIS IS THE EXTRA1 STREAM ***
Header length: 33
Stream flags: 00000000
```

```
Stream name: EXTRA2
Stream size: 1024 bytes
Maximum lines: 100
Lines per IDB: 5
Stream type: 0
Stream header: *** THIS IS THE EXTRA2 STREAM ***
Header length: 33
Stream flags: 00000000

Stream name: EXTRA3
Stream size: 1024 bytes
Maximum lines: 100
Lines per IDB: 5
Stream type: 0
Stream header: *** THIS IS THE EXTRA3 STREAM ***
Header length: 33
Stream flags: 00000000
```

The parameters that define the streams are located at approximately '19A4'X in ADFMDFLT. The label on the stream definitions is ADFIDSTR. The data at that location is a full-word initialized to '00000009', indicating nine streams are defined for the Session Manager.

Each stream is defined via eight full-words that follow the stream number. The data is as follows:

1. Stream name pointer: This word points to an eight byte field containing the name of the stream. The name is left justified and padded with blanks.

2. Stream size: This word contains a hexadecimal number indicating the length of the stream. This number represents the total number of bytes available in the stream for text. A stream will wrap if the maximum number of bytes has been reached and the stream is defined as one that can wrap. The stream flags determine the ability of a stream to wrap. Refer to the Session Manager Program Logic Manual (form LY28-0913) for more information regarding stream wrapping.

   Stream management overhead can be determined by dividing the number of lines per IDB (IDB's are described below) into the maximum number of lines in the stream and adding one to the result.

   Let's work an example using the SMOUT stream. It is defined as 4096 bytes long, 150 lines maximum and 5 lines per IDB. Using the above verbal formula results in the following:

$$\frac{150}{5} = 30 + 1 = 31 \text{ IDB's}$$

The 31 IDB's can actually index 155 lines of data and using QUERY.STREAMS with the IBM supplied defaults would show this.

Since IDB's are eight bytes long, the overhead is:

31 IDB's X 8 bytes per IDB = 248 bytes overhead

Thus, 248 bytes of stream management overhead is obtained in addition to the text portion size. The two physical core requirements really define one logical stream consisting of an overhead area and a text area.

3. Maximum lines: This word contains a hexadecimal number indicating the total number of lines that may be entered into the stream. If this number is exceeded, the stream will wrap or overlay previous stream entries. Note that a stream will wrap if the maximum number of lines is reached even if the byte capacity of that stream has not been exceeded.

4. Lines per IDB: This word contains a hexadecimal number indicating the number of logical lines for which an IDB (index block) "is responsible". The IDB's are part of an indexing mechanism the Session Manager uses for referencing stream data. As described earlier, each IDB is eight bytes long.

5. Stream type: This word is initialized to one of the following values to define the type of stream:

   - 0 - defines an EXTRA stream type. In the IBM supplied defaults, the HEADER, EXTRA2, and EXTRA3 streams are of this type.
   - 1 - defines an INPUT stream type. In the IBM supplied defaults, the TSOIN and SMIN streams are of this type.
   - 2 - defines an OUTPUT stream type. In the IBM supplied defaults, the TSOOUT, SMOUT, MESSAGE, and EXTRA1 streams are of this type.

   The stream types are almost self-explanatory. The EXTRA and EXTRA2 streams are not actively used by the Session Manager. Optionally, the user may direct data to an EX-

TRA stream via Session Manager commands. The INPUT streams are used for TSO command input (TSOIN) and Session Manager input (SMIN). The OUTPUT streams are used for any output generated by TSO commands (TSOOUT), Session Manager commands(SMOUT) and output from SNAPSHOT commands (EXTRA1). The MESSAGE stream is not actively used until it is targeted for use via Session Manager commands.

6. Stream header: The stream header is a 32 or 33 character string that becomes the first record in the stream it is associated with.

   The character string's main use is for ease of stream identification. When SCROLLing through a stream, the header identifies the name of the stream.

7. Header length: This word is initialized (in hexadecimal) to the length of the stream header.

8. Stream flags: This word is defined as a flag field. Currently, only two bits are defined - bit 0, if on, indicates the stream does not wrap and bit 1, if on, indicates the alarm should sound when data is entered into the stream. The remaining thirty(30) bits are reserved.

Changing Stream Definitions

As mentioned earlier, the stream definitions are located in module ADFMDFLT at approximately '19A4'X in the module at label ADFIDSTR. The stream definitions may be altered using program IMASPZAP, if an installation deems it necessary. Every stream may be altered in some fashion but, it is important to remember the following rules:

- The Session Manager has a fixed number of streams. That number is nine and the full-word at label ADFIDSTR in module ADFMDFLT (initialized to '00000009') should never be changed. Unpredictable results will occur if the number is changed.

- The Session Manager requires the following stream names - TSOIN, TSOOUT, SMIN, SMOUT, EXTRA1, EXTRA3, and HEADER. The stream size, lines per IDB and stream header may be changed but, the stream names, stream type, and stream flags of these required streams must not be changed.

Let's discuss some potential changes and their implications.

Changing the stream size obviously means more or less storage will be obtained for the particular stream being changed. This would allow more data to be entered into a stream (if the size is increased) or less data to be entered (if the size is decreased). The TSOIN stream will probably wrap if the default stream size is not changed. Increasing the size to hopefully hold all TSO commands and data a user enters during a session obviously has a penalty - more storage would be required. Decreasing the size would cause the stream to wrap sooner. The penalty would be loss of some stream data by overlaying or wrapping.

Changing the maximum number of lines in a stream also has an affect on the frequency of wrapping. If the maximum number of lines is changed to a high number, a stream could wrap because the number of bytes entered exceeded the stream capacity but wasn't close to the maximum number of lines. If the maximum number of lines is set quite low, the stream could wrap with a large amount of stream capacity unused.

The "lines per IDB", as defined in the default, are set at five for all streams. The number of IDB's created, based on formulas discussed earlier, can be decreased by increasing the number of lines per IDB. The net result would show a decrease in the amount of overhead (less storage for IDB's) but increased search time when SCROLLing through a stream. Decreasing the number of lines per IDB, increases the total number of IDB's and storage required for them. But, the efficiency of stream accessing improves when there are fewer lines per IDB.

The defaults supplied by IBM, stream sizes, maximum lines and lines per IDB, provide a level of performance that should prove to be acceptable to most installations. If not, by following the above guidelines, changes can easily be made.

The remaining parameters (stream type, stream header, stream flags) may be changed as long as one additional precaution is exercised:

- The stream headers should remain the same length as currently defined.

One final note to emphasize a prior important point:

- The stream type and stream flags of the required streams (TSOIN, TSOOUT, SMIN, SMOUT, EXTRA1, EXTRA3 and HEADER) must not be changed or unpredictable results will occur.

Creating a Default Environment Module

The previous two sections assumed that you would be using IMASPZAP to alter the IBM supplied default environment module ADFMDFLT. Now suppose you want to provide multiple default environments for your TSO users. For each environment you want, you can provide a logon procedure that specifies a different defaults module to be used by the Session Manager.

IBM supplies a copy of the assembler source of ADFMDFLT in SYS1.SAMPLIB(ADFDFLTX), so you may modify it, reassemble it, and create a new load module for another default environment or you may choose to code your own default environment module.

The next section describes the format required for a defaults module. The section after that is an example of how the assembler source of ADFMDFLT might be modified to provide an environment for a 3278 Model 4 display terminal (i.e., 43 lines by 80 columns).

Format of a Default Environment Module

A default environment module may contain up to three CSECTs. The default environment csect must be the entry point. The other two CSECTs may be installation exit routines. (See "Installation Management Exit Routines").

The default environment CSECT contains four basic areas:

1. A header that contains some required data and addresses at fixed offsets into the module.

2. A table of stream definitions.

3. A table of function definitions.

4. A table of Session Manager commands that are stacked into the SM input stream to initialize the screen layout and PF key definitions.

Now let's go into more detail about each of these areas.

1. The header is 116 bytes of information at the beginning of the module. The first 8 bytes must be the characters 'ADFMDFLT'. This is to ensure that the defaults csect is the entry point in the load module. If any other csect is the entry point, then the end user will get an error message and will get TSO without the Session Manager.

   The second 8 bytes may be a time stamp or zeros.

The next 76 bytes are not used. (In the IBM supplied ADFMDFLT, this area is used for a copyright statement).

Starting at hex offset 5C are the addresses of the installation management exit routines you may have link-edited into this module. These eight bytes must be zero if no exits are there.

Starting at hex offset 68 are the addresses of the three tables in the defaults csect.

```
OFFSET     LABEL
           ADFMDFLT      CSECT
  0                      DC CL8'ADFMDFLT'
  8                      DC CL8'&SYSDATE
 10                      DS CL76
 5C                      DC V(inst-exit-1)
 60                      DC V(inst-exit-2)
 64                      DS CL4
 68                      DC AL4(ADFIDCMD)
 6C                      DC AL4(ADFIDSTR)
 70                      DC AL4(ADFIDFUN)
```

2. The table of stream definitions begins with a fullword containing the number of streams. The remainder of the table contains contiguous 32-byte entries, one entry for each stream. Each entry contains the following information:

- pointer to the stream name
- number of bytes in the stream
- maximum number of lines in the stream
- number of lines per IDB
- stream type
- length of stream header line
- pointer to the stream header line
- flags for stream attributes

```
 LABEL
ADFIDSTR      DC A(# streams)

stream        DC AL4(stream-name-addr)
entry#1       DC A(# bytes)
              DC A(# lines)
              DC A(# lines/IDB)
              DC A(stream-type)      0=extra
                                     1=input
                                     2=output
              DC A(header-length
              DC AL4(header-addr)
              DC BL4('xx0...0')      1x...=non-wrapable
                                     x1...=alarm is on
```

You must provide entries in this table for TSOIN, TSOOUT, SMIN and SMOUT since some Session Manager commands have ope-

rands that default to these particular stream names. Furthermore, if you intend to use the basic IBM supplied defaults for the screen layout and PF key definitions, then you must also have table entries for EXTRA1, EXTRA3, and HEADER. You may add other stream definitions as you please.

See the previous two sections about what to consider when defining the number of bytes, number of lines, and number of lines per IDB.

The actual stream names and stream header lines pointed to by this table may be located anywhere in the defaults csect as long as they are after the header and not within any of the three tables.

3. The table of function definitions begins with a fullword containing the number of functions. The remainder of the table contains contiguous 28-byte entries, one entry for each function. Each entry contains the following information:

   - pointer to the function name
   - pointer to the input stream name for this function
   - pointer to the output stream name for this function
   - intensity of output lines
   - pointer to the copy stream name for this function
   - intensity of copied lines
   - flags for function attributes

```
LABEL
ADFIDFUN     DC A(# functions)

function     DC AL4(function-name-addr)
entry #1     DC AL4(input-stream-name-addr)
             DC AL4(output-stream-name-addr)
             DC A(output-intensity)    0=non-display
                                       1=normal
                                       2=high
             DC AL4(copy-stream-name-addr)
             DC A(copy-intensity)      0=non-display
                                       1=normal
                                       2=high
             DC BL4('xx0...0')      1x...=alarm on output
                                    x1...=alarm on input
```

The Session Manager supports three functions and you must provide entries in this table for each. The first entry must be TSO, the second must be SM, and the third must be MSG. You may add function definitions if you like, but the Session Manager can support them only with the CHANGE.FUNCTION and QUERY.FUNCTION commands.

The actual function names and stream names pointed to by this table may be located anywhere in the defaults csect as

long as they are after the header and not within any of the three tables.

4. The table of Session Manager command strings begins with a fullword containing the number of command strings, followed by a fullword containing the total number of bytes taken up by all command string entries. (This number does not include the first two fullwords of the table). The remainder of the table contains contiguous 4-byte entries, one entry for each command string. Each entry contains the following information:

   - length of command string
   - pointer to the command string

```
            LABEL
ADFIDCMD    DC F'#entries'
            DC F'#bytes'

command     DC F'#bytes'
entry #1    DC AL4(command-string-addr)
```

   The Session Manager will pick up each of these command strings and stack them into the SM function's input stream during initialization. Once the Session Manager is initialized, these commands will be executed.

   You may put an entry in this table for any valid Session Manager command to define the screen layout, define PF keys, refine the definitions of streams, functions, terminal characteristics, or anything else Session Manager commands allow the end user to do.

   Note: If you put nothing in this table, then the end user will have no screen layout to enter commands. The only thing available would be the CLEAR key which will allow the entry of one Session Manager command.

   The actual command strings pointed to by this table may be located anywhere in the defaults csect as long as they are after the header and not within any of the three tables.

Example of a Default Environment Module

IBM supplies the assembler source of the defaults module ADFMDFLT. If you are coding your own defaults module, then you might consider making a copy of this source and using it as a base.

Suppose you want to create an environment for the 3278 Model 4 display terminal with the window MAIN taking up the additional

lines on the larger screen. (I.E., this window will be 38 lines instead of 19). The other windows will remain at the lower portion of the screen.

The following is an outline of what the defaults module would look like once the required modifications were made to the IBM supplied source of ADFMDFLT:

Note:

1. An asterisk (*) in the left margin indicates a changed line (the modified characters are underlined).

2. Offsets are shown only for the header.

```
        OFFSET   LABEL
                 ADFMDFLT CSECT
          00             DC CL8'ADFMDFLT'
          08             DC CL8'&SYSDATE'
          10             DC CL76'...'
          5C             DC V(exit-routine1-addr)
          60             DC V(exit-routine2-addr)
          64             DS CL4
          68             DC AL4(ADFIDCMD)
          6C             DC AL4(ADFIDSTR)
          70             DC AL4(ADFIDFUN)
           :     ADFCC1   DC CL26'...'
           :     ADFCC2   DC CL46'...'
                   :
                   :
      *          ADFCC12  DC CL111'DEFINE.WINDOW MAIN
                                  1  1  38  80  ...'
      *          ADFCC13  DC CL98'DEFINE.WINDOW LINE
                                 39  1  1  80  ...'
      *          ADFCC14  DC CL116'DEFINE.WINDOW CURRENT
                                 40  1  2  62  ...'
      *          ADFCC15  DC CL115'DEFINE.WINDOW STITLE
                                 40  63  1  12  ...'
      *          ADFCC16  DC CL115'DEFINE.WINDOW SVALUE
                                 40  75  1  6  ...'
      *          ADFCC17  DC CL115'DEFINE.WINDOW LTITLE
                                 41  63  1  9  ...'
      *          ADFCC18  DC CL114'DEFINE.WINDOW LVALUE
                                 41  72  1  9  ...'
      *          ADFCC19  DC CL113'DEFINE.WINDOW ENTRY
                                 42  1  1  120  ...'
      *          ADFCC20  DC CL99'DEFINE.WINDOW VLINE
                                 43  41  1  2  ...'
      *          ADFCC21  DC CL115'DEFINE.WINDOW PASSWD
                                 43  43  1  38  ...'
                   :
                   :
                 ADFCC53  DC CL43'...'
```

```
ADFSN1   DC CL8'TSOIN'
ADFSH1   DC CL32'***THIS IS THE TSOIN STREAM***'
ADFSN2   DC CL8'TSOOUT'
ADFSH2   DC CL33'***THIS IS THE ...'
   :
   :
ADFSN9   DC CL8'SMIN'
ADFSH9   DC CL32'***THIS IS THE ...'
ADFFN1   DC CL4'TSO'
ADFFIN1  DC CL8'TSOIN'
ADFFON1  DC CL8'TSOOUT'
ADFFCN1  DC CL8'TSOOUT'
ADFFN2   DC CL4'SM'
ADFFIN2  DC CL8'SMIN'
ADFFON2  DC CL8'SMOUT'
ADFFCN2  DC CL8'SMOUT'
ADFFN3   DC CL4'MSG'
ADFFON3  DC CL8'TSOOUT'

ADFIDCMD DC F'53'
         DC F'5577'

         DC F'26'
         DC AL4(ADFCC1)
         DC F'46'
         DC AL4(ADFCC2)
         :
         :
         DC F'43'
         DC AL4(ADFCC53)

ADFIDSTR DC A(9)
         DC AL4(ADFSN1)
         DC A(8192)
         DC A(300)
         DC A(5)
         DC A(1)
         DC A(32)
         DC AL4(ADFSH1)
         DC BL4'00...00'
         :
         :
         DC AL4(ADFSN9)
         DC A(8192)
         DC A(300)
         DC A(5)
         DC A(1)
         DC A(32)
         DC AL4(ADFSH9)
         DC BL4'00...00'

ADFIDFUN DC A(3)
         DC AL4(ADFFN1)
         DC AL4(ADFFIN1)
```

```
                        DC AL4(ADFFON1)
                        DC A(1)
                        DC AL4(ADFFCN1)
                        DC A(2)
                        DC BL4'00...00'
                        :
                        :
                        DC AL4(ADFFN3)
                        DC AL4(ADFFIN3)      ZERO
                        DC AL4(ADFFON3)
                        DC A(2)
                        DC AL4(ADFFCN3)      ZERO
                        DC A(0)
                        DC BL4'10...0'
```

Note:

1. ADFCC12 through ADFCC21 are modified to rearrange the locations of the windows on the screen, making the window MAIN larger.

2. In the ADFIDFUN table entry for the MSG function, there are pointers to the names of the input and copy streams. These pointers are zero since the MSG function has no input and copy streams.

3. ADFCC34 through ADFCC45 are the command strings to define 24 PF keys. The deinitions for PF keys 1-12 are identical to these for PF keys 13-24. If the environment is being created for use with terminals having 24 PF keys, then you can redefine either set of twelve keys. Just remember to make the oppropriate modifications to the command string table ADFIDCMD also.

Installation Management Exit Routines

The Session Manager supports installation exits to allow the installation to monitor the terminal user's conversation with the system.

One exit is supported by the Session Manager initialization program (ADFMDF01) to allow the installation to indicate what Session Manager streams are to be monitored for this user's session.

Another exit is supported by the program that puts a line into a stream (ADFIMPUT). Whenever a line is to be put into a stream, the Session Manager determines if the stream is being monitored. If it is, then the exit routine is called and the line, the name of the stream, a timestamp, and other information are passed. The line is not put into the stream until after the exit routine is called.

Depending upon how and where you supply these installation exits you can provide a wide range of monitoring capabilities. You may want to monitor all users the same way; you may want to monitor each class of users differently; or you may want some combination of monitoring most users one way and monitoring certain classes of users other ways.

The next three sections discuss the following:

- how and where these exits are installed and how the Session Manager calls them
- the interface between the Session Manager and the exit routines
- some programming considerations

Location of Installation Exits

At least one pair of installation exits must be supplied to provide any monitoring capability. There are two places where each of the exit routines can be placed.

1. One or both exits may be link-edited into the ADFMDF01 load module. The exit to be called during Session Manager initialization must be named ADFEXIT1 and the other must be named ADFEXIT2.

2. One or both exits may be link-edited into any of the default environment modules you have. These exits may be named anything, but offset '5C'X into the defaults csect must contain the address of the exit to be called during Session Manager initialization and offset '60'X must contain the address of the exit called by ADFIMPUT.

If there is an initialization exit routine in the defaults module, then it will override ADFEXIT1. Likewise, if the exit routine to be called by ADFIMPUT is in the defaults module, then it will override ADFEXIT2.

There is no restriction as to the pair of exits being in the same load module. You may provide ADFEXIT1 and ADFEXIT2 in ADFMDF01, then for some default environments, override one or both of them with different exits to tailor the monitoring of certain classes of users. You may choose to have no ADFEXIT1 and/or no ADFEXIT2 also. However, no monitoring of users can take place if one of the pair of exits is not supplied in one of the two load modules.

The following is a scenario of how the Session Manager finds the addresses of the exit routines during initialization.

1. If the fullword at offset '5C'X in the defaults csect is non-zero, then use it as the address of the initialization exit routine and GOTO step 4.

2. If the address of ADFEXIT1 is non-zero, then use ADFEXIT1 and GOTO step 4.

3. There is no initialization exit, so monitoring cannot be done. GOTO END.

4. If the fullword at offset '60'X in the defaults csect is non-zero, then use it as the address of the exit routine called by ADFIMPUT and GOTO END.

5. If the address of ADFEXIT2 is non-zero, then use ADFEXIT2 and GOTO END.

6. There is no exit to be called by ADFIMPUT, so monitoring cannot be done.

END.

Note: The addresses of the exits to be called are kept in a Session Manager control block. Once these two addresses are initialized, the load module locations of the exits are transparent to the Session Manager. For subsequent discussion, let's assume the exits to be called are ADFEXIT1 and ADFEXIT2.

Installation Management Exit Routines Interfaces

This section describes parameter lists and register contents upon entry to the installation exits.

ADFEXIT1 is called by the Session Manager initialization program ADFMDF01. This exit is used to instruct the Session Manager as to what streams are to be monitored for this user.

The register contents upon entry to ADFEXIT1 are:

```
R0      - unpredictable
R1      - addr (parameter list)
R2:R12  - unpredictable
R13     - addr (ADFMDF01 save area)
R14     - return address
R15     - addr (ADFEXIT1)
```

The parameter list pointed to by register 1 contains:

+0: Pointer to the userid (The userid is left justified and padded with blanks).

+4: Pointer to installation data (The installation data is one fullword of zeros. This fullword is reserved for the installation's use. The Session Manager will preserve its contents across all calls to ADFEXIT2).

+8: Pointer to a bit mapping (This mapping is one byte long and the various bits represent the streams that can be monitored. This byte is initially zero and the various bits are set by ADFEXIT1 to indicate which streams are to be monitored).

```
1xxx x...--> TSO input stream
x1xx x...--> TSO output stream
xx1x x...--> SM input stream
xxx1 x...--> SM output stream
xxxx 1...--> MSG output stream
```

Note that particular stream names are not monitored, but the input/output streams of particular functions are monitored. In other words, if ADFEXIT1 sets the first bit, then the TSO function's input stream will be monitored. This is not necessarily the TSOIN stream since the end user may have issued the CHANGE.FUNCTION command to make TSO's input stream something else.

ADFEXIT1 receives control in supervisor state under protection key 1 and holding no locks. ADFEXIT1 must ensure that control is returned to the Session Manager the same way.

ADFEXIT2 is called by the Session Manager routine ADFIMPUT that puts a line into a stream. ADFIMPUT determines if the target stream of the line being written is one of the monitored streams. If so, then ADFEXIT2 is called.

The register contents upon entry to ADFEXIT2 are:

```
R0      - unpredictable
R1      - addr (parameter list)
R2:R12  - unpredictable
R13     - addr (ADFIMPUT save area)
R14     - return address
R15     - addr (ADFEXIT2)
```

The parameter list pointed to by register 1 contains:

+0: Pointer to the userid (The userid is left justified and padded with blanks).

+4: Pointer to installation data (The installation data is one fullword reserved for the installation's use. This fullword will contain whatever ADFEXIT1 put there on whatever the last call to ADFEXIT2 put there. The Session Manager will preserve its contents across calls to the exit routines).

+8: Pointer to a bit mapping (This mapping is one byte long and indicates to which stream the line is being written. Note that this mapping is not the same one passed to ADFEXIT1 and will have one and only one bit set).

```
1xxx x...--> TSO input stream
x1xx x...--> TSO output stream
xx1x x...--> SM input stream
xxx1 x...--> SM output stream
xxxx 1...--> MSG output stream
```

Note again that these bits do not represent particular stream names. They represent only the input or output stream of a particular function.

+C: Pointer to a time stamp (This time stamp is the two fullwords returned by a STORE CLOCK instruction executed by ADFIMPUT. The 31st bit of the first word represents 1.048576 seconds).

+10: Pointer to length of control data (This length is a two byte field representing the number of Session Manger control characters in the line being written to the stream. This length may be zero).

+14: Pointer to length of data (This length is a two byte field representing the number of text characters being written to the stream).

+18: Pointer to control data (Currently, the Session Manager provides up to one byte of control data).

```
1x.. ...x--> line is displayed with high intensity
x1.. ...x--> line is non-display
xx.. ...1--> line is to be treated as TPUT ASIS
```

This control data indicates to the Session Manager special characteristics of the line that must be handled when the line is to be viewed by a window on the screen.

- The first two bits will cause the line to be displayed at something other than normal intensity.

- If the last bit is on, then the Session Manager performs only minimal editing on the line before displaying it on the screen. (I.E., the same editing as TPUT ASIS is done.) Normally, the Session Manager performs the same editing as TPUT EDIT. (For a more detailed description of this editing, see Guide to Writing a Terminal Monitor Program or a Command Processor, GC28-0648).

+1C: Pointer to data (The data is the text of the line being written to the stream.

ADFEXIT2 receives control in supervisor state under protection key 1 and holding the local lock. ADFEXIT2 must not release the local lock. The local lock is used by the Session Manager for serialization and releasing it will cause unpredictable results.

The length fields, control data, and text passed to ADFEXIT2 constitutes the actual line being processed by ADFIMPUT. ADFEXIT2 must not change the length fields as unpredicatable results may occur. However, ADFEXIT2 can change the control data and/or the text as long as the lengths remain the same. ADFIMPUT will do no validity checking of these parameters upon return from ADFEXIT2.

Programming Considerations for Installation Exits

This section describes some programming considerations for the installation management exits.

- providing different exit routines for different classes of users
- examples of the kinds of things the installation can do with these exits.

As described earlier in the section "Location of Installation Exits", you can tailor the monitoring capabilities of the exits right down to a particular class of users who use the same logon procedure. The following are some considerations to further illustrate this capability:

- Besides ADFEXIT1 and ADFEXIT2, you may supply as many pairs of exit routines as you have default environment modules.
- ADFEXIT1 and ADFEXIT2 may be supplied to provide a global monitoring capability of all Session Manager users. These exits may then be overridden for a particular class of users by simply supplying different exit routines in the defaults module that is used by that class of users.
- If you do not want to monitor all users, then do not supply ADFEXIT1 and ADFEXIT2. Supply exit routines only in the defaults modules used by the classes of users you do want to monitor.

The following are a few examples of the kinds of things you may do with these exit routines and some special considerations:

1. There may be certain commands that should not be issued by a particular user or class of users. If you are monitoring TSO input and the exit intercepts one of these commands, then the command can be changed to blanks or some other characters, effectively discarding the command. However,

remember that the length of the line must not be changed.

2. Keep in mind that when a TSO input line is intercepted, it is merely being placed in the TSO input stream. It may not be ready for execution yet since other commands may be stacked ahead of this one. Therefore, if the exit routine is to perform some special action based upon a particular command being entered, then it does not suffice to merely detect the command being placed in the TSO input stream. The command should be detected as it is copied to the TSO output stream. It is at that time the Session Manager is in the process of returning that line of input to TSO for execution.

3. It seems somewhat inefficient to have to monitor every line of TSO output if all the exit routine is interested in is intercepting a particular command being copied to the output stream. A more effecient method is to monitor only TSO input until this particular command is detected. At that time the exit routine can begin monitoring TSO output until the command is copied to the output stream. Some special action may then be taken and the exit routine may go back to monitoring only TSO input. Dynamically changing what streams are being monitored can be done by simply having ADFEXIT1 save the address of the stream mapping as a piece of installation data. This stream mapping that defines what streams are being monitored may then be changed by ADFEXIT2.

   If this is your chosen implementation, then there are some special considerations to keep in mind.

   - Once the command is detected being copied to the TSO output stream, if the exit routine simply turns off the monitoring of TSO output, then it may miss other occurrences of the command that have already been entered into the TSO input stream. (For example, the user may have typed the command more than one time on the screen and hit ENTER once). You may want to have a counter associated with each command and the counter can be incremented by one when the command is intercepted as input and decremented by one when the command is copied to the output stream. If decrementing results in the counter going to zero, then monitoring of TSO output can be turned off at that time.

   - If the exit routine is looking for a READY mode message followed by a particular command, then there may be complications if the user's TSO output and MSG output streams are the same and the user happens to receive a message just after the READY mode message. (See #8 below).

4. Keep in mind that ADFEXIT2 is being called by ADFIMPUT, the Session Manager program that is called each time a line is

to be put into any stream. Although this is a mainline path, there is a negligible increase in the number of instructions in ADFIMPUT to determine if ADFEXIT2 is to be called and to call it. However, you should consider the following:

- Depending upon what streams you are monitoring, ADFEXIT2 could possibly be called every time through ADFIMPUT. If this happens and ADFEXIT2 has a considerable amount of function, then there could be a performance degradation.

- As described earlier, ADFEXIT2 can dynamically change the mapping that defines what streams are to be monitored. This can be done to reduce the number of unnecessary calls to ADFEXIT2.

- If one group of users is to be monitored differently, then the overhead of monitoring one group does not have to be incurred upon the other group. Different sets of exit routines can be supplied in the groups' default environment modules. (See "Location of Installation Exits").

5. Suppose you want ADFEXIT2 to retain a log of a user's TSO session. This can be done by intercepting each line to the TSO output stream and writing it to a data set. When the LOGOFF command is intercepted being copied to the output stream, the exit routine can hardcopy the data set.

   It is recommended, however, that such I/O not be done within ADFEXIT2 since it could cause performance degradation. As an alternative, ADFEXIT1 can attach another task during initialization to handle I/O requests from ADFEXIT2. Whenever ADFEXIT2 is ready to write a record, it can post the I/O task and pass it a buffer with the record. ADFEXIT2 can then return to ADFIMPUT, allowing the Session Manager to continue while the I/O task is synchronously writing the record.

   Note also that the task attached by ADFEXIT1 will run without holding the local lock, so ADFEXIT2 never has to release it. Keep in mind that releasing the local lock in ADFEXIT2 will cause unpredictable serialization problems within the Session Manager.

6. The Session Manager goes through the following steps to determine if ADFEXIT2 should be called:

   a. If no streams are being monitored or the address of ADFEXIT2 is zero, then GOTO END.

   b. If TSO input is being monitored and the target stream=TSO input stream, then call ADFEXIT2 and GOTO

END.

c. If TSO output is being monitored and the target stream=TSO output stream, then call ADFEXIT2 and GOTO END.

d. If MSG output is being monitored and the target stream=MSG output stream, then call ADFEXIT2 and GOTO END.

e. If SM input is being monitored and the target stream=SM input stream, then call ADFEXIT2 and GOTO END.

f. If SM input is being monitored and the target stream = SM input stream, then call ADFEXIT2 and GOTO END.

END.

If the user has the same stream defined as the input or output stream of two or three functions, then the following can happen:

- ADFEXIT2 will be called for EVERY line going to a monitored stream, even if the line is being put into the stream as the result of another function. (For example, if MSG output is being monitored and the user has TSOOUT as both the TSO and MSG output stream, then ADFEXIT2 will be called for all TSO output, since all such lines have a target stream equal to the MSG output stream).

- The bit mapping passed to ADFEXIT2 will indicate only the stream for which a match was found between the target stream and the function's stream. (In the last example, the bit mapping would always indicate MSG output, even though many of the lines were TSO output lines).

7. A knowledgeable terminal user could fabricate the entire TSO output stream by issuing Session Manager commands to put the expected lines into the TSO output stream. The exit routine can save all TSO input in a data set for subsequent verification of what TSO commands were actually issued, if this is a potential problem.

8. The time stamp provided with each line passed to ADFEXIT2 can be useful for monitoring the execution time of particular commands or for providing a time stamp on lines saved for subsequent hardcopy listings.

- Suppose a particular command is to be monitored to sample system performance. ADFEXIT2 can monitor TSO input and upon intercepting the command, begin monitoring TSO output. When the command is copied to the TSO output stream, the time stamp can be saved and when ADFEXIT2 detects the READY mode message, it can calculate the difference between the saved time stamp and the time

stamp provided with the READY mode message line to determine the execution time of the command.

- If users' TSO sessions are being saved for subsequent hardcopy, then the time stamp passed to ADFEXIT2 for each line can be converted to a printable time stamp and written with each line. Anyone reviewing this session could then see at what time certain operations were being performed.

TGET/TPUT Extended Parameter Lists

The Session Manager uses an extended TGET/TPUT parameter list. These lists provide a mechanism for getting or putting lines of text from or to Session Manager streams. (They are currently used by the Session Manager TSO commands SMCOPY, SMFIND, and SMPUT).

The Session Manager intercepts all TGETs and TPUTs and when the I/O request happens to be one with this extended parameter list, then the Session Manager gets or puts a line from or to a specified stream.

You can write programs to utilize these extended parameter lists. A program may dynamically redefine the screen layout and PF keys by TPUTing the necessary Session Manager commands to the SM input stream. (These commands will be executed while processing the TPUT). TSO commands may be TPUT to the TSO input stream. (These commands will be executed when TSO is ready for input).

For specific details of these extended parameter lists, see the Session Manager Program Logic Manual (LY28-0913).

6. MESSAGES

The error and informational messages issued by the Session Manager are listed in this section. Messages having id's ending in "A" require action on the part of the user; those ending in "I" are informational only.

ADF001A PRESS ENTER KEY TO BEGIN SESSION MANAGER/TSO SESSION
The Session Manager is ready to assume control.

ADF002I SYNTAX ERROR ON SESSION MANAGER LOGON PROCEDURE
The PARM field contained invalid or unrecognized values.

ADF003I SESSION MANAGER HAS ABNORMALLY TERMINATED. CONTINUING WITHOUT THE SESSION MANAGER.
The Session Manager has terminated because of an internal error. The TSO session will continue without the Session Manager. Re-LOGON to activate the Session Manager.

ADF004I SESSION MANAGER TERMINATED - ERROR RETURN CODE FROM
ESTAE (SVC 60)
TPUT (SVC 93)
TGET (SVC 93)
ATTACH OF MONITOR TASK(ADFMDF01)
ATTACH OF TERMINAL MONITOR PROGRAM
The Session Manager encountered an error while invoking a system function. The severity of the error prevented continuation of the session.

ADF005I MODULE module-name SPECIFIED ON LOGON PROCEUDRE NOT FOUND
The Session Manager could not find the default-environment-module specified as a parameter on the EXEC statement of the logon procedure. The TSO session will continue without the Session Manager.

ADF005A PRESS ENTER KEY TO CONTINUE WITHOUT SESSION MANAGER
The message is issued after ADF005I and ADF006I.

ADF006I ERROR IN MODULE module-name SPECIFIED ON LOGON PROCEDURE
The default-environment-module specified as a parameter on the EXEC statement of the logon procedure was found and loaded by the Session Manager, but the format of the load module was incorrect. (The CSECT containing the default definitions must be the entry point of the load module and must begin with the character string 'ADFMDFLT'). The TSO session will continue without the Session Manager.

ADF010I COMMAND command NOT FOUND
The command entered does not exist.

ADF011I INVALID COMMAND MODIFIER: modifier
The modifier entered had an invalid form or was incorrect for the command.

ADF012I INVALID COMMAND SYNTAX: command entered
The command entered was of invalid form.

ADF013I INVALID OPERAND(S): operand
The operand entered had an invalid form or was not an operand for the command.

ADF014I MISSING OPERAND(S): command entered
A required operand was not found.

ADF015I TOO MANY OPERANDS: command entered
More operands were entered than are acceptable at one time.

ADF016I COMMAND MODIFIER MISSING: command entered
A required command modifier was not found.

ADF017I INVALID SCROLL AMOUNT: value entered
The window could not scroll the number of lines, pages or columns specified or the command was syntactically incorrect.

ADF018I ERROR IN SUBFIELD OF operand OPERAND
The subfield entered was incorrect (for example, a window name that does not exist) or unexpected (too many subfields) or of invalid form.

ADF019I MISSING SUBFIELD FOR operand OPERAND
A required subfield for the operand was not found.

ADF021I WINDOW window-name NOT FOUND
The window-name entered does not exist. Use QUERY.WINDOWS to verify window names.

ADF022I STREAM stream-name NOT FOUND
The stream-name entered does not exist. Use QUERY.STREAMS to verify stream names.

ADF023I TEXT text-string NOT FOUND
The text-string searched for by the FIND command was not located.

ADF024I LINE NUMBER line-number NOT FOUND
The stream line number specified in the SCROLL.ABSOLUTE command does not exist.

ADF025I STREAM stream-name IS FULL
The named stream is of the non-wrapping type, and all available space has been filled or the input line was larger than the stream.

ADF027I WINDOW window-name COULD NOT BE DEFINED
The window to be defined overlapped another window on the screen, or the maximum allowable number of windows was reached.

ADF028I ERROR DURING read from (or) write to TERMINAL BY SESSION MANAGER
Invalid data was received from the terminal (possibly from an unformatted screen) or a bad return code was received from TGET/TPUT.

ADF029I SESSION MANAGER INTERNAL ERROR IN module-name
An unexpected internal error occurred in a Session Manager routine.

ADF030I WINDOW window-name ALREADY EXISTS
The window name being defined already exists. Use QUERY.WINDOWS to verify window names.

ADF031I window-name VIEWING LINE nnnnnn
The top of the specified window is viewing line number nnnnnn.

ADF040A ENTER SESSION MANAGER COMMAND(S):
The user has pressed the CLEAR key. At this time only Session Manager commands can be entered. Pressing the PA2 key will restore the screen to display the contents prior to CLEARing the screen.

ADF041A ENTER A NULL LINE TO RETURN TO FULL SCREEN PROGRAM
A command running under SPF Option 6 has completed and, in order to return to SPF full-screen support, a null line must be entered.

ADF101I SMCOPY INTERNAL LOGIC ERROR - SMCOPY TERMINATED
This indicates a serious internal error has occurred. If issued, a coding error in SMCOPY was encountered.

ADF102I INPUT SOURCE DATA SET COULD NOT BE OPENED - SMCOPY TERMINATED
A system function (OPEN) failed when trying to perform an operation on the source data set. The copy operation could not be performed.

ADF103I OUTPUT TARGET DATA SET COULD NOT BE OPENED - SMCOPY TERMINATED
A system function (OPEN) failed when trying to perform an operation on the target/output data set. The copy operation could not be performed.

ADF104I INPUT SOURCE DATA SET HAS INVALID RECORD FORMAT - SMCOPY TERMINATED
The source data set must have fixed or variable length record format. Contact your systems programmer for assistance.

ADF105I OUTPUT TARGET DATA SET HAS INVALID RECORD FORMAT - SMCOPY TERMINATED
The target data set must have fixed or variable length record format. Refer to the SMCOPY command description or contact your systems programmer for assistance.

ADF106I ERROR DURING WRITE TO OUTPUT DATA SET - SMCOPY TERMINATED
An I/O error was detected during the copy operation. SMCOPY is terminated at that point.

ADF107I ERROR DURING READ FROM INPUT DATA SET - SMCOPY TERMINATED
An I/O error was detected during the copy operation. SMCOPY is terminated at that point.

ADF108I ERROR DURING WRITE TO OUTPUT SESSION MANAGER STREAM - SMCOPY TERMINATED
An error was detected during the copy operation. SMCOPY is terminated at that point.

ADF109I ERROR DURING READ FROM INPUT SOURCE SESSION MANAGER STREAM - SMCOPY TERMINATED
An error was detected during the copy operation. It is ter-

minated at that point.

ADF110I SESSION MANAGER NOT ACTIVE - SMCOPY IGNORED
The Session Manager is not active and the SMCOPY command is ignored. To activate the Session Manager, re-LOGON using a Session Manager LOGON procedure.

ADF111I OUTPUT DATA SET NOT FOUND - ASSUMED TO BE NEW
The user specified a data set as target for the copy. It did not exist, so SMCOPY creates one. The copy operation continues.

ADF112I INPUT STREAM NAME NOT FOUND - SMCOPY TERMINATED+
The source Session Manager stream specified could not be found. Use QUERY.STREAMS to verify stream names.

ADF112A USE THE SESSION MANAGER "QUERY" COMMAND TO DISPLAY VALID STREAM NAMES
Friendly advice to the user.

ADF113I OUTPUT STREAM NAME NOT FOUND - SMCOPY TERMINATED+
The target Session Manager stream specified could not be found. Use QUERY.STREAMS to verify stream names.

ADF113A USE THE SESSION MANAGER "QUERY" COMMAND TO DISPLAY VALID STREAM NAMES
Friendly advice to the user.

ADF114I RECORD FORMAT OF DATA SET(S) NOT CONSISTENT WITH OPERATION BEING PERFORMED
Data sets must have fixed or variable length record format. Refer to the SMCOPY command description or contact your systems programmer for assistance.

ADF115I FROM DATA SET MEMBER NOT GIVEN - TEMPNAME ASSUMED
A Partitioned Data Set was specified as the "FROM" data set but no member was given. Member "TEMPNAME" will be used if found. Otherwise, a prompt for a new data set name will occur.

ADF116I FROM DATA SET MEMBER NOT FOUND
The specified member in the "FROM" data set does not exist. A prompt for a new data set name will occur.

ADF117I TO DATA SET MEMBER NOT GIVEN - TEMPNAME ASSUMED
The specified "TO" data set is a Partitioned Data Set but no member was specified. "TEMPNAME" is the assumed member.

ADF118I FROM DATA SET ORGANIZATION INVALID
The data set organization of the FROM data set is not one of the required types. The only acceptable data set organization is PO (partitioned data set) or PS (physical sequential).

ADF119I TO DATA SET ORGANIZATION INVALID
The data set organization of the TO data set is not one of the required types. The only acceptable data set organization is PO (partitioned data set) or PS (physical sequential).

ADF201I TEXT FOUND AT LINE line-number
The text specified in the SMFIND command exists at the line specified.

ADF202I TEXT NOT FOUND
The text specified in the SMFIND command was not located in the stream.

ADF203I SESSION MANAGER STREAM stream-name DOES NOT EXIST+
The stream specified in the SMFIND command could not be found. Use QUERY.STREAMS to verify stream names.

ADF203A USE THE SESSION MANAGER "QUERY" COMMAND TO DISPLAY VALID STREAM NAMES
Friendly advice to the user.

ADF204I SESSION MANAGER NOT ACTIVE - SMFIND IGNORED
The Session Manager is not active and the SMFIND command is ignored. To activate the Session Manager, re-LOGON using a Session Manager LOGON procedure.

ADF205I ERROR DURING READ WHILE SEARCHING STREAM - SMFIND TERMINATED
An error occurred using a system function, SMFIND could not continue.

ADF301I ERROR DURING WRITE TO SESSION MANAGER STREAM
An error occurred using a system function, SMPUT could not continue.

ADF302I SESSION MANAGER STREAM stream name DOES NOT EXIST+
Stream name entered does not exist. Use QUERY.STREAMS to verify stream names.

ADF302A USE THE SESSION MANAGER "QUERY" COMMAND TO DISPLAY VALID STREAM NAMES
Friendly advice to the user.

ADF303I SESSION MANAGER NOT ACTIVE - SMPUT IGNORED
The Session Manager is not active and the command is ignored. To activate the Session Manager, re-LOGON using a Session Manager LOGON procedure.

APPENDIX A: DEFAULT PF KEY DEFINITIONS

The Program Function keys, as defined in the IBM-supplied default display environment are shown here in a reduced size so that you may cut out or copy this page for use on your terminal keyboard.
| A program function key template is also available, GX20-1981.

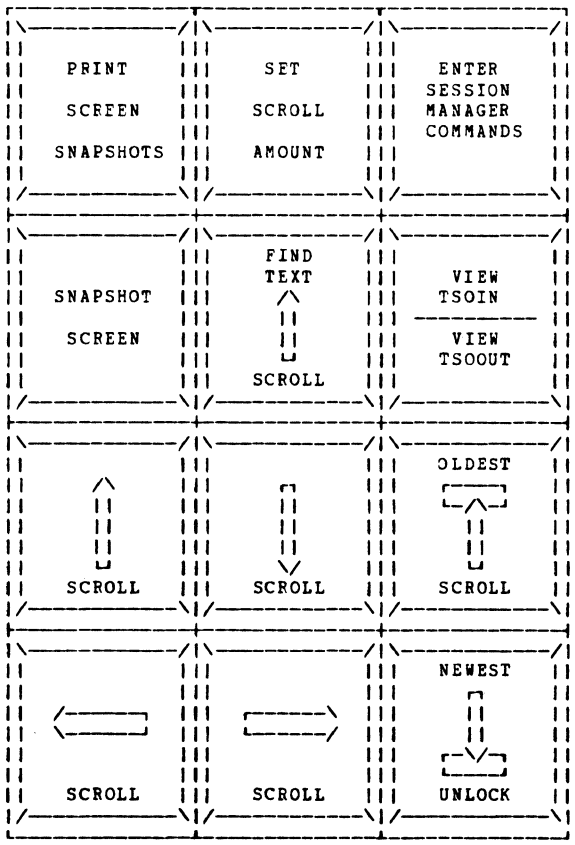


Figure A.1 - Default PF Key Definitions

Environment Definition Commands

CHANGE.CURSOR C C	[row column] [window-name] [TEMPORARY]
CHANGE.FUNCTION C F	function [ALARM(INPUT | OUTPUT | NO)] [COPY(copy-stream-name [intensity]) | NOCOPY] [INPUT(input-stream-name)] [OUTPUT(output-stream-name [intensity])]
CHANGE.PFK C P	pfk-number definition-text-string target-stream-name [SUBSTITUTE[(identifier [delimiter])]]
CHANGE.STREAM C S	stream-name [ALARM(YES | NO)] [CLEAR]
CHANGE.TERMINAL C T	[ALARM(YES | NO)] [CONTROL(LAST | seconds)]

```
                    [DEFAULT(window-name)]


CHANGE.WINDOW       [window-name]
C      W
                    [ALARM(YES | NO)]

                    [HOLD(INPUT | seconds)]

                    [OVERLAP(lines)]

                    [PROTECT(YES | NO)]

                    [TARGET(stream-name [intensity])]

                    [UPDATE(LINE | NEWEST | PAGE)]

                    [VIEW(stream-name)]
```

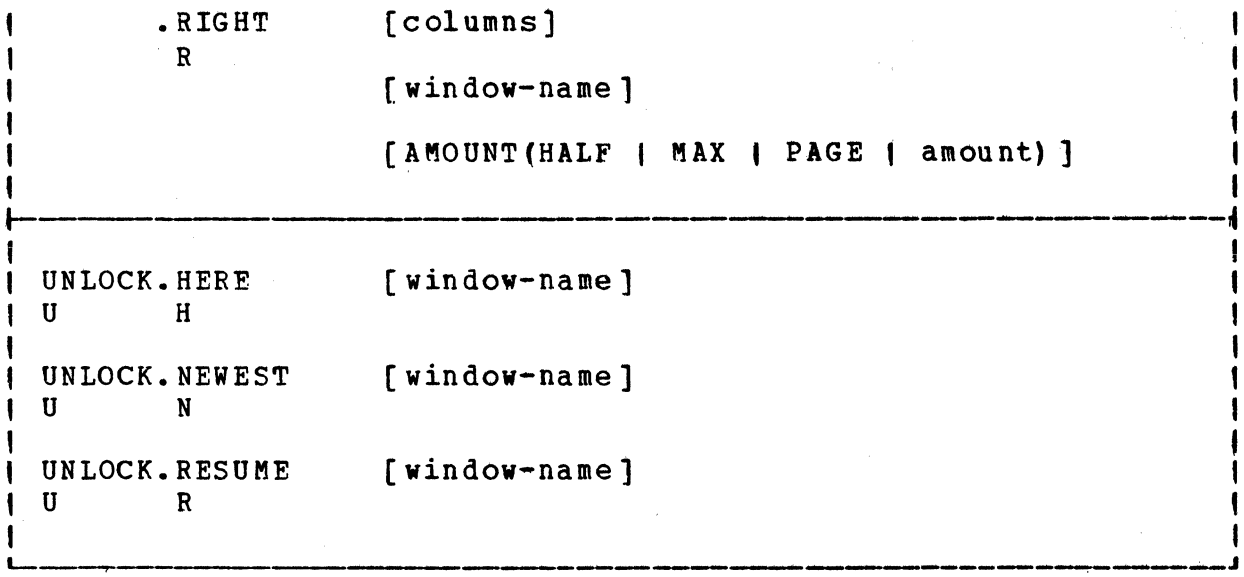
```
DEFINE.WINDOW       window-name
D      W
                    row

                    column

                    lines

                    width

                    [ALARM(YES | NO)]

                    [HOLD(INPUT | seconds)]

                    [OVERLAP(lines)]

                    [PROTECT(YES | NO)]

                    [TARGET(stream-name [intensity])]

                    [UPDATE(LINE | NEWEST | PAGE)]

                    [VIEW(stream-name)]
```

```
DELETE.WINDOW       window-name | *
DEL    W
```

Screen Control Commands

```
FIND.BACKWARD      text-string
F    B
                   [window-name]

FIND.FORWARD       text-string
F    F
                   [window-name]

FIND.LINE          [window-name]
F    L
                    TARGET(stream-name)
```

```
SCROLL.ABSOLUTE    line-number
S      A
                   [window-name]


       .BACKWARD   [pages]
        B
                   [lines]

                   [window-name]

                   [AMOUNT(HALF | MAX | PAGE | amount)]

       .FORWARD    [pages]
        F
                   [lines]

                   [window-name]

                   [AMOUNT(HALF | MAX | PAGE | amount)]

       .LEFT       [columns]
        L
                   [window-name]

                   [AMOUNT(HALF | MAX | PAGE | amount)]

       .NEWEST     [window-name]
        N

       .OLDEST     [window-name]
        O
```


.RIGHT
R
[columns]
[window-name]
[AMOUNT(HALF | MAX | PAGE | amount)]
UNLOCK.HERE
U H
[window-name]
UNLOCK.NEWEST
U N
[window-name]
UNLOCK.RESUME
U R
[window-name]

Session Control Commands

```
END

PUT                 text-string
P
                    target-stream-name

                    [INTENSITY(intensity)]

QUERY.FUNCTIONS     [target-stream-name]
Q     F

      .PFKS         [target-stream-name]
       P

      .STREAMS      [target-stream-name]
       S

      .TERMINAL     [target-stream-name]
       T

      .WINDOWS      [target-stream-name]
       W

RESET

RESTORE.PFKS
R       P

        .SCREEN
         S

        .WINDOW     [window-name]
         W

SAVE.PFKS
SA   P

     .SCREEN
      S

     .WINDOW        [window-name]
```

```
      W
```

```
SNAPSHOT          target-stream-name
SN
                  [FORMAT]
```

TSO Commands

```
SMCOPY [FROMDATASET(dsname) |
SMC     FROMSTREAM(stream-name) ]

       [PRINT(sysout-class) |
        TODATASET(dsname) |
        TOSTREAM(stream-name) ]

       [ASIS | CAPS | NOTRANS]

       [FORMAT | NOFORMAT| PREFORMAT]

       [LINE(start-line[:stop-line]) ]
```

```
SMFIND text-string
SMF
       [STREAM(stream-name) ]

       [BACKWARD | FORWARD]

       [ALL | FIRST]

       [ANY | ASIS]

       [LINE(line1[:line2]) ]
```

```
SMPUT text-string
SMP
       [target-stream-name]

       [INTENSITY(intensity) ]
```

OS/VS2 MVS TSO
3270 Extended Display Support -
Session Manager
User's Guide and Reference
SC28-0912-0

READER'S COMMENT FORM

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. This form may be used to communicate your views about this publication. They will be sent to the author's department for whatever review and action, if any, is deemed appropriate.

IBM shall have the nonexclusive right, in its discretion, to use and distribute all submitted information, in any form, for any and all purposes, without obligation of any kind to the submitter. Your interest is appreciated.

Note: ***Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.***

Possible topics for comments are:

Clarity Accuracy Completeness Organization Coding Retrieval Legibility

If comments apply to a Selectable Unit, please provide the name of the Selectable Unit _____.

If you wish a reply, give your name and mailing address:

_____

_____

_____

Note: Staples can cause problems with automated mail sorting equipment. Please use pressure sensitive or other gummed tape to seal this form.

— Cut or Fold Along Line —

Please circle the description that most closely describes your occupation.

Customer	(Q) Install Mgr.	(U) System Consult.	(X) System Analyst	(Y) System Prog.	(Z) Applica. Prog.	(F) System Oper.	(I) I/O Oper.	(L) Term. Oper.	(O) Other

IBM	(S) System Eng.	(P) Prog. Sys. Rep.	(A) System Analyst	(B) System Prog.	(C) Applica. Prog.	(D) Dev. Prog.	(R) Comp. Prog.	(G) System Oper.	(J) I/O Oper.	(E) Ed. Dev. Rep.	(N) Cust. Eng.	(T) Tech. Staff Rep.

Number of latest Newsletter associated with this publication: _____

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments.)

SC28-0912-0

Reader's Comment Form

Fold and tape | Please Do Not Staple | Fold and tape

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

International Business Machines Corporation
Department D82, Building 706-2
PO Box 390
Poughkeepsie, New York 12602

Fold and tape | Please Do Not Staple | Fold and tape

IBM

Cut or Fold Along Line

OS/VS2 MVS TSO 3270 Extended Display Support - Session Manager User's Guide and Reference Printed in U.S.A. SC28-0912-0

OS/VS2 MVS TSO
3270 Extended Display Support
Session Manager
User's Guide and Reference
SC28-0912-0

READER'S COMMENT FORM

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. This form may be used to communicate your views about this publication. They will be sent to the author's department for whatever review and action, if any, is deemed appropriate.

IBM shall have the nonexclusive right, in its discretion, to use and distribute all submitted information, in any form, for any and all purposes, without obligation of any kind to the submitter. Your interest is appreciated.

Note: ***Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.***

Possible topics for comments are:

Clarity Accuracy Completeness Organization Coding Retrieval Legibility

If comments apply to a Selectable Unit, please provide the name of the Selectable Unit _____.

If you wish a reply, give your name and mailing address:

_____

_____

_____

Note: Staples can cause problems with automated mail sorting equipment. Please use pressure sensitive or other gummed tape to seal this form.

Cut or Fold Along Line

Please circle the description that most closely describes your occupation.

Customer	(Q) Install Mgr.	(U) System Consult.	(X) System Analyst	(Y) System Prog.	(Z) Applica. Prog.	(F) System Oper.	(I) I/O Oper.	(L) Term. Oper.					(O) Other
IBM	(S) System Eng.	(P) Prog. Sys. Rep.	(A) System Analyst	(B) System Prog.	(C) Applica. Prog.	(D) Dev. Prog.	(R) Comp. Prog.	(G) System Oper.	(J) I/O Oper.	(E) Ed. Dev. Rep.	(N) Cust. Eng.	(T) Tech. Staff Rep.	

Number of latest Newsletter associated with this publication: _____

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments.)

SC28-0912-0

Reader's Comment Form

Cut or Fold Along Line

OS/VS2 MVS TSO 3270 Extended Display Support - Session Manager User's Guide and Reference Printed in U.S.A. SC28-0912-0

Fold and tape | Please Do Not Staple | Fold and tape

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

International Business Machines Corporation
Department D82, Building 706-2
PO Box 390
Poughkeepsie, New York 12602

Fold and tape | Please Do Not Staple | Fold and tape

IBM®

Technical Newsletter

This Newsletter No. SN28-2928
Date June 30, 1978

Base Publication No. SC28-0912-0
Prerequisite Newsletters None

OS/VS2 MVS TSO 3270 Extended Display Support - Session Manager User's Guide and Reference

This Newsletter contains replacement pages for *Session Manager User's Guide and Reference* to support Session Manager (5740-XE2).

Before inserting any of the attached pages into *Session Manager User's Guide and Reference* read **carefully** the instructions on this cover. They indicate when and how you should insert the pages.

Pages to be Removed	Attached Pages to be Inserted
Cover, Edition Notice	Cover, Edition Notice
1 - 8	1 - 8
43 - 44	43 - 44
53 - 54	53 - 54
57 - 60	57 - 60
81 - 82	81 - 82
97 - 100	97 - 100
105 - 108	105 - 108
115 - 116	115 - 116
145 - 150	145 - 150
153 - 160	153 - 160.2
165 - 166	165 - 166
169 - 180	169 - 180

A change to the text or to an illustration in indicated by a vertical line to the left of the change.

Summary of Amendments

Additions and changes have been made in this publication to support Release 2 of OS/VS2 MVS TSO 3270 Extended Display Support - Session Manager.

Note: *Please file this cover at the back of the base publication to provide a record of changes.*

IBM Corporation, Publications Development, Department D58, Building 706-2, PO Box 390, Poughkeepsie, New York 12602

Printed in U.S.A.

OS/VS2 MVS TSO 3270 Extended Display Support - Session Manager User's Guide and Reference Printed in U.S.A. SC28-0912-0

SC28-0912-0